DOCUMENT RESUME

ED 247 391 CE 039 402

AUTHOR	Litecky, Charles R.; Lamkin, Tim
TITLE	Data Processing (Advanced Business Programming) Volume II. Instructor's Guide.
INSTITUTION	Missouri Univ., Columbia. Instructional Materials Lab.
PUB DATE	May 84
NOTE	805p.; For related documents, see ED 220 074 and ED 220 604. Printed on colored paper.
PUB TYPE	Guides - Classroom Use - Guides (For Teachers) (052)
EDRS PRICE	MF05/PC33 Plus Postage.
DESCRIPTORS	Behavioral Objectives; *Business; Computers; *Computer Science; *Computer Software; Course Descriptions; Curriculum Guides; *Data Processing; Data Processing Occupations; Learning Activities; Lesson Plans; Office Occupations Education; Postsecondary Education; *Programing; *Programing Languages; Teaching Guides; Transparencies

ABSTRACT

This curriculum guide for an advanced course in data processing is for use as a companion publication to a textbook or textbooks; references to appropriate textbooks are given in most units. Student completion of assignments in Volume I, available separately (see ED 220 604), is a prerequisite. Topics covered in the 18 units are introduction, documentation and design tools, systems analysis and design, program design, nested and other IF statements, control breaks, multiple-level control breaks, additional COBOL elements, sequential files, sorting for sequential files I and II, business data processing in BASIC, Report Program Generator (RPG) programming (RPG II business application programs and RPG II tables and arrays), data entry and edit programs, and general business reports. Each unit may contain some or all of the following: unit and specific objectives, suggested instructor and student activities, a list of required instructional materials, notes to the instructor, information sheets in outline form, handouts, transparency masters, assignment sheets, answers to assignment sheets, job sheets, unit examination, and answers to the examination. Three types of activities are provided: hands-on programming, paper exercises, and reading or lecture. Computers that support COBOL and/or RPG programs are required for student completion of assignment and job sheets. (YLB)

****
* Reproductions supplied by EDRS are the best that can be made *
* from the original document. *
****

ED247391

INSTRUCTOR'S GUIDE

DATA PROCESSING
(ADVANCED BUSINESS PROGRAMMING)
VOLUME II

Charles R. Litecky, Ph.D., CDP
Associate Professor
College of Business and Public Administration
University of Missouri-Columbia

Tim Lamkin, President, CEO
RPG Units
Lamkin and Associates, Inc.
Columbia, Missouri

Produced by the Instructional Materials Laboratory
University of Missouri-Columbia
Columbia, Missouri

Catalog Number BOE-17-I
May 1984

CE 039402

U.S. DEPARTMENT OF EDUCATION
NATIONAL INSTITUTE OF EDUCATION
EDUCATIONAL RESOURCES INFORMATION
CENTER (ERIC)
This document has been reproduced as received from the person or organization originating it.
Minor changes have been made to improve reproduction quality.

- Points of view or opinions stated in this document do not necessarily represent official NIE position or policy.

"PERMISSION TO REPRODUCE THIS
MATERIAL HAS BEEN GRANTED BY

TO THE EDUCATIONAL RESOURCES
INFORMATION CENTER (ERIC)."

FOREWORD

This guide, Data Processing, Volume II, completes a series of three curriculum publications. The first curriculum guide in the series, Introduction to Data Processing, is designed for a one-semester course, and the instructor may use microcomputers in the offering or elect not to use any hardware. Data Processing, Volumes I and II are for full-year course offerings. Therefore, it requires two years to complete both volumes. The learning activities are written in a generic format so that they can be adapted to various types of hardware.

The subject matter content in Volume II is more advanced, and it is assumed that a student has completed the assignments in Volume I before proceeding to the learning activities in Volume II. Computers which support COBOL and/or RPG programs will be needed by the trainees to complete the assignment and job sheets. It is suggested that the design and documentation units be covered before any other learning activities. Practical COBOL application programs involving report breaks, sequential, and index sequential file updating for payroll are included in Units VI, XI, and XIII.

RPG instructional materials are furnished in four major units. Units XV and XVII provide the student with program exercises based on concepts learned in Volume I. These exercises are designed to move from simple to complex tasks as the student progresses through the learning activities. Unit XVI presents two advanced topics in RPG--Tables and Arrays. Unit XVIII introduces the student to the benefits of data processing and includes a discussion of business applications that are appropriate for small to moderate-sized business.

COBOL, RPG, and BASIC languages are presented in both volumes. The RPG units can be covered independently of COBOL or BASIC. BASIC probably will be easier for the student and might be covered before COBOL or RPG. The instructor may find it helpful to pull the units on one language from each volume and combine them when planning a course.

Although business and office examples are used throughout this three-guide series, the information is applicable to any course offering instruction on the use of computers.

ACKNOWLEDGMENTS

Appreciation is extended to those individuals who contributed their time and talents to the development of this curriculum guide.

For providing special assistance:

Mr. Charles A. Newman, Director
Business and Office Education
Department of Elementary and Secondary Education
Jefferson City, MO 65102

Dr. Mary Jane Lang
Professor of Education
University of Missouri-Columbia
Columbia, MO 65211

For planning and reviewing the guide content:

Ms. Sharon Fredd
Hannibal AVTS

Mr. Tim Lamkin
Carrollton AVTS

Mr. Don Marsh
Nichols Career Center, Jefferson City

Ms. Mary Ann Penniston
Hillyard AVTS, St. Joseph

Mrs. Carol Spry
Herndon AVTS, Raytown

For technical assistance from the Instructional Materials Laboratory, University of Missouri-Columbia, 10 Industrial Education Building, Columbia, MO 65211

Harley Schlichting, Director
Amon Herd, Associate Director
Phyllis Miller, Editor
Dan Stapleton, Graphic Designer
Crystal Dietiker, Graphic Layout Specialist
Paul Hughes, Commercial Artist
Mary Brunstrom, Composition Typist
Kelly Clark, Composition Typist

DATA PROCESSING

VOLUME II

TABLE OF CONTENTS

Foreword iii

Acknowledgments v

How to Use This Guide xviii

UNIT I: INTRODUCTION 1

HO 1: Example of Experience-Oriented Resume 15

TM 1: Overview: The Programming Process 17
TM 2: Organization of a Typical DP Department 18
TM 3: DP Job Types 19
TM 4: Major Programming Job Markets 20

AS 1: Visit a Computer Installation 21
AS 2: Mock Interview 22
AS 3: Prepare Resume 23

Unit II: DOCUMENTATION AND DESIGN TOOLS 29

HO 1: Program Flowchart Example 39

TM 1: Data Flow Diagram Example 41
TM 2: Visual Table of Contents Example 42
TM 3: HIPO Example of an Overview Diagram 43
TM 4: HIPO Relationships Between Diagrams 44
TM 5: Most-Used Program Flowcharting Symbols 45
TM 6: Warnier-Orr Diagram Example 46
AS 1: Summary and Evaluation of Techniques 47

UNIT III: SYSTEMS ANALYSIS AND DESIGN 55

HO 1: Example Results of the Preliminary Study 67
HO 2: Example Results of the Detailed Investigation 68
HO 3: Types of Controls 69

TM 1: DP and Company Organization 71
TM 2: The Systems Life Cycle 72
TM 3: Analysis of Collected Facts in Systems Analysis Phase 73

AS 1: The Preliminary Results Report 75
AS 2: System Control Design 76

UNIT IV: PROGRAM DESIGN 83

TM 1: Example of a Functionally Cohesive Module 93
TM 2: Example of a Module with Improper Coincidental Coupling 94
TM 3: Problems in Common Coupling 95
TM 4: Control Coupling Example 96
TM 5: Module Size Rules 97

AS 1: General Review of Program Design 99
AS 2: Design Exercise 100

UNIT V: NESTED AND OTHER IF STATEMENTS 107

TM 1: Multiple Level Nested IF's with Inverted Pairing 121
TM 2: Example Programming Specifications for Writing a Nested IF 122
TM 3: The Most Inclusive Condition 123
TM 4: Summary of Steps in Writing Nested IF's 124
TM 5: Use of Submodule to Simplify Complex IF Statements 125
TM 6: Example of a Complex IF Simplified Through Parenthesis 126

AS 1: Simple Nested IF's with Matched and Unmatched ELSE's 127
AS 2: Semi-matched and NEXT SENTENCE IF Sentences 128
AS 3: Multiple Level IF Sentences 129
AS 4: Designing Nested IF's 130
AS 5: Combined and Complex IF Statements 131

UNIT VI: CONTROL BREAKS 145

HO 1: The Program Plan - Programming Specifications 157
HO 2: Pseudocode for Control Inventory Report Module 158
HO 3: Pseudocode Specifications for Detail Record Processing Module 159
HO 4: Pseudocode Specifications for Processing Department Changes Module 160

HO 5: Coding for Control Inventory Report Module 161

TM 1: Example of Control Break Input 163
TM 2: Report Subtotal Examples of Printing During Control Breaks 164
TM 3: Example Card Format for a Control Break Program for a Simple Inventory 165
TM 4: Initial Processing Steps in Control Break Processing 166
TM 5: Processing of First Detail Record in Control Break Programs 167
TM 6: Processing of a Detail Record in Control Break Programs 168
TM 7: Processing an Actual Control Break in Control Break Programs 169
TM 8: Final Total Processing 170
TM 9: Input Record Format for Sample Control Break Program 171
TM 10: Program Output for Sample Control Break Program 172
TM 11: IPO Chart and Hierarchy Chart for the Sample Program 173
TM 12: IPO Chart for Process Detail Records Module 175
TM 13: Hierarchy Chart for Process Detail Records Module 176
TM 14: IPO Chart for Process Department Changes Module 177
TM 15: IPO Chart for Print the Final Totals Module 178
TM 16: IPO Chart for Print the Headings Module 179
TM 17: Pseudocode Specifications for Final Total Printing 180
TM 18: Pseudocode for Printing the Headings 181

AS 1: Terms and Background Information on Control Breaks 183
AS 2: Initial Detail and Second Detail Processing in Control Break Programs 184
AS 3: Actual Control Break Processing and Final Total Control Break Processing 185
AS 4: A Sample Control Break Program 186

JS 1: Writing a Payroll Validation Report Using Control Break Logic 189

Fig. JS 1-1: Input Card Layout for Payroll Validation Report 191
Fig. JS 1-2: Sample Control Hierarchy Chart for Payroll Validation Report Program 192

Fig. JS 1-3: Prewritten Divisions for Payroll Validation Report 193

JS 1: Appendix: Sample Application, an Inventory List by Branch Location Report Application 195

Fig. JS 1-4: Input Layout for Inventory by Branch Location 197
Fig. JS 1-5: Hierarchy Chart for Inventory Report by Branch Location Program 198
Fig. JS 1-6: Program Listing of Sample Application for Inventory by Branch (Parts 1 and 2) 199
Fig. JS 1-7: Sample Output of Inventory Listing by Branch Location (Parts 1 and 2) 201

UNIT VII: MULTIPLE-LEVEL CONTROL BREAKS 211

HO 1: Programming Specifications 223
HO 2: Pseudocode for Control Sales Report Module 224
HO 3: Pseudocode Specifications for Process Detail Records Module 225
HO 4: Pseudocode Specifications for Process Customer Change Module 226
HO 5: Pseudocode Specifications for Process Salesman Change Module 227
HO 6: Pseudocode Specifications for Process Branch Change Module 228

TM 1: Multiple-Level Control Break Report with Group Indication 229
TM 2: Printer Spacing Chart and Input Record Format 230
TM 3: Processing the First Card as a Detail Record 231
TM 4: IPO Chart and Hierarchy Chart for Sample Program 232
TM 5: IPO Chart - Process Detail Records Module 233
TM 6: IPO Chart - Process Customer Change Module 234
TM 7: IPO Chart - Process Salesman Change Module 235
TM 8: IPO Chart - Process Branch Change Module 236
TM 9: IPO Chart - Print the Final Total Module 237
TM 10: Group-Indicated Report Sample 238
TM 11: COBOL Coding - Control Sales Report Module 239
TM 12: COBOL Coding - Process Detail Record Module 240
TM 13: Source Listing (Part 1 and Part 2) 241

AS 1: Terms and Introduction to Multiple-Level Control Breaks 243
AS 2: Overview of Sample Program, Initial, and Detailed Processing 244
AS 3: Actual Processing of the Four Levels of Control Breaks 245
AS 4: IPO Charts for Sample Program Modules 246
AS 5: Pseudocode for Sample Program Modules 247

UNIT VIII: ADDITIONAL COBOL ELEMENTS 255

TM 1: Example of Renames Clause 265
TM 2: Example of Name Qualification 266
TM 3: Example of GO TO DEPENDING Statement 267
TM 4: General Formats of INSPECT Statement 268
TM 5: Example of INSPECT Statement (Part 1 and Part 2) 269

AS 1: Level 77 and Level 66 Entries 271
AS 2: Name Qualification, BLANK WHEN ZERO, and Figurative Constants 272
AS 3: GO TO, ACCEPT/DISPLAY, and ON SIZE ERROR Statements 273

UNIT IX: SEQUENTIAL FILES I 281

HO 1: Program Specifications (Parts 1, 2, 3, and 4) 291
HO 2: Pseudocode for Edit Estimated Hours Module 295
HO 3: Source Listing of Sample Program (Parts 1, 2, 3, and 4) 296

TM 1: Creating a Sequential Master File 301
TM 2: Application Program Master File Information 302
TM 3: Example of Check Totals (Parts 1 and 2) 303
TM 4: IPO Chart for Top-level Module 305
TM 5: Overall Hierarchy Chart 306
TM 6: Definition of Sequential File on Disk 307
TM 7: Example of the Use of Subscripted Fields (Parts 1, 2, and 3) 308
TM 8: Transaction File Input Record 311
TM 9: Example of Checking for Missing Data 312

AS 1: Edit Checks 313
AS 2: IPO Charts Summary 314

UNIT X: SORTING FOR SEQUENTIAL FILES 321

HO 1: Program Specifications (Parts 1 and 2) 329

TM 1: Example of Report Produced from Two Input Files 331

TM 2: Format of Employee Identification File 332
TM 3: Application Programmer Status Report Format 333
TM 4: Environment Division Entries for Sort Files 334
TM 5: Data Division File Definitions (Part I and 2) 335
TM 6: Example of Sort Verb 337
TM 7: Create Status Report Module 338
TM 8: Hierarchy Chart 339
TM 9: Example of Matching of Records Procedure 340
TM 10: Pseudocode Specifications - Obtain Status Report Data 341
TM 11: Source Listing (Part I and 2) 342

UNIT XI: SEQUENTIAL FILE UPDATING 349

TM 1: Flow Diagram of Sequential Update Application 361
TM 2: Edit and Sort Programs Preceding a Typical Sequential File Update 362
TM 3: Format of Program Master File 363
TM 4: Foremat of Transaction Record 364
TM 5: Program Specifications (Parts 1, 2, 3, and 4) 365
TM 6: IPO Chart - Update Program Master File 369
TM 7: Final Hierarchy Chart 370
TM 8: Pseudocode for Update Program Master Files 371
TM 9: Pseudocode for Obtain New Master Data 372
TM 10: Definition of Old and New Program Master Files 373
TM 11: Source Listing (Parts 1, 2, and 3) 374

AS 1: Introduction to Sequential File Updating 377
AS 2: General Logic of Sequential File Update Programs 378
AS 3: Flow chart of the General Logic of a Sequential Master File Update 379
AS 4: Tracing the Flow of Records through Sequential File Update Logic 380

JS 1: Writing a Complete Sequential File Update for Payroll (with an Appendix) 383

- Fig. JS 1-1: Payroll Application To Create, Update, and Print Payroll Master Files 387
- Fig. JS 1-2: Prewritten Divisions for Payroll Applications 388
- Fig. JS 1-3: Pseudocode Specifications for Program Applications 391
- Fig. JS 1-4: Input Files for Use in H1, H2, and H3 COBOL Programs 392

Fig. JS 1-5: Steps for Testing H1 393
Fig. JS 1-6: Output of H1 394
Fig. JS 1-7: Hierarchy Chart for Payroll Sequential File Update Program, H2 395
Fig. JS 1-8: Pseudocode for Payroll Sequential File Update Program, H2 (Parts 1 and 2) 397
Fig. JS 1-9: Output from Program H2 398
Fig. JS 1-10: Output from Program H3 399

JS 1: Appendix: Sample Application, Sequential Update Program for Bibliography 401

Fig. JS 1-11: Hierarchy Chart for Sample Bibliography Update Program 404
Fig. JS 1-12: Listing of Sample File Update Program 405
Fig. JS 1-13: File Layout and Listing of Old Master File For Sample Program 406
Fig. JS 1-14: Transaction Card Layout and List of Transactions Used in Sample Run for Sample Bibliography Update Program 409
Fig. JS 1-15: Output for Sample Run of Biblio - Update Program Combined with File Contents (Parts 1 and 2) 410
Fig. JS 1-16: Bibliography Update Transaction Report 412

UNIT XII: INDEXED SEQUENTIAL FILES I 423

HO 1: Pseudocode for Get Valid Pair of Records Module 435
HO 2: Pseudocode Master File for Write Customer Module 436
HO 3: Source Listing of Sample Indexed Sequential Program (Parts 1, 2, 3, 4, and 5) 437

TM 1: Format of Customer Master File 443
TM 2: Format of Input Records 444
TM 3: Format of the Exception Report 445
TM 4: Format of the Customer Master File Listing 446
TM 5: Overview of Sample Program to Load Indexed Sequential File 447
TM 6: IPO Chart - Load Customer Master File 448
TM 7: Hierarchy Chart for Sample Program 449
TM 8: Select Sentence - Indexed Sequential File 450
TM 9: Environment and Data Division Entries for Sequential Retrieval 451

TM 10: File Definition of Customer Master File 452

AS 1: Understanding the Sample Program Specifications 453

UNIT XIII: INDEXED SEQUENTIAL FILES II 461
HO 1: Complete Hierarchy Chart 475
HO 2: Pseudocode Specifications (Part 1 through 12) 476
HO 3: Source Listing (Part 1 through 7) 488

TM 1: Overview of Random Update 495
TM 2: Format of Customer Master File 496
TM 3: Format of Customer Transaction Input File 497
TM 4: Format of Transaction Report 498

AS 1: Understanding the Sample Program 499

JS 1: Writing A Complete Indexed Sequential Program for Payroll 503

- Fig. JS 1-1: Overview of Application to Create, Update, and Print Indexed Sequential Payroll Master Files 506
- Fig. JS 1-2: Hierarchy Chart for Payroll Indexed Sequential File Update Program, H2 507
- Fig. JS 1-3: Pseudocode Indexed Sequential File Update H2 508
- Fig. JS 1-4: Prewritten Dividions for Indexed Sequential Payroll Application 513
- Fig. JS 1-5: Output from Program H2 515
- Fig. JS 1-6: Output Showing Debugging and Updating Traces (Parts 1 and 2) 516

JS 1: Appendix: Sample Application Indexed Sequential Update Program for Inventory 519

- Fig. JS 1-7: VSAM Setup Program for an Inventory Application 520
- Fig. JS 1-8: Listing of Sample Indexed Sequential File Update Program for Inventory 522

UNIT XIV: BUSINESS DATA PROCESSING IN BASIC 541

HO 1: Pseudocode for Main Menu Program, "RMENU.BAS" 557
HO 2: Source Listing for Main Menu Program, "RMENU.BAS" 558
HO 3: Pseudocode for "RNEWCUS.BAS" Program to Add New Customers to Accounts Receivable File for Rogers Company 559
HO 4: Source Listing for the New Customer Program, "RNEWCUS.BAS" 560
HO 5: Source Listing for the Sales and Payments Program, "RSALPAY.BAS" 561
HO 6: Pseudocode for "RPRINT.BAS", Program to Print Accounts Receivable 562

TM 1: Typical Application of Microcomputers 563
TM 2: An Example Menu 564
TM 3: The Rogers Company Accounts Receivable System 565
TM 4: Sample Design of the Main Menu Screen 566
TM 5: Design of the Screen for Addition of New Customers 567
TM 6: Hierarchy Chart for the New Customer Program, "RNEWCUS.BAS" 568
TM 7: Design of the Sales and Payments Screen 569
TM 8: Hierarchy Chart for the Sales and Payments Program 570
TM 9: Overview of the Print Statements Program 571

AS 1: BASIC File Commands 573
AS 2: BASIC Buffer Commands 574

JS 1: Print Program for Rogers Company Accounts Receivable 577

UNIT XV: RPG II BUSINESS APPLICATION PROGRAMS 585

HO 1: Overview of Assignment Sheets 1, 2, 3, and 4 595
HO 2: Overview of Accounts Receivable Register Problem 596
HO 3: Overview of Master Subscriber File Problem 597
HO 4: Overview of Electric Bill Calculations Problem 599
HO 5: Overview of Employee Payroll Deductions 600

AS 1: Accounts Receivable Register 603
AS 2: Master Subscriber File Update 604
AS 3: Electric Bill Calculations 606
AS 4: Employee Payroll Deductions 609

JS 1: Execute Accounts Receivable Register on Computer 623
JS 2: Execute Master Subscriber File Program 624
JS 3: Execute Electric Bill Problem 625
JS 4: Execute Payroll Calculation Program 626

UNIT XVI: RPG II TOPICS: TABLES AND ARRAYS 631

HO 1: Table Look-up Example 645
HO 2: Calculations and GOTO Operation 646
HO 3: Two Related Tables 647
HO 4: Sample Table Problem and Coded Solution 648
HO 5: Example of an Array 650
HO 6: Using XFOOT and MOVEA 651
HO 7: Indexing with Arrays 652
HO 8: Using a Field as the Index 653

TM 1: Problem System Flowchart 655
TM 2: Types of Tables and Arrays 656
TM 3: Table and Array Data Operations 657
TM 4: Examples of XFOOT and MOVEA 658

AS 1: Sample Assignment Sheet Program 659
AS 2: Coding Extension Sheets I 661
AS 3: Coding Extension Sheets II 662
AS 4: Describing a Pre-execution-time Table 663
AS 5: Searching a Table 664
AS 6: Finding a Value in a Table 665
AS 7: Array Problem and Questions 667

UNIT XVII: DATA ENTRY AND EDIT PROGRAMS 675

TM 1: Data Processing Routine 681

AS 1: Program 1 Description 683
AS 2: Program 2 Description 691

UNIT XVIII: GENERAL BUSINESS REPORTS 725

HO 1: Sample Invoice 741
HO 2: Sample Invoice Edit List 742
HO 3: Sample Invoice/Sales Register 743
HO 4: Sample Aged Trial Balance Report 744
HO 5: Sample Customer Statement 745
HO 6: Sample Delinquency Notice 747
HO 7: Sample Stock Status Review 748
HO 8: Sample A-B-C Analysis Report 749
HO 9: Sales Analysis and Profit Analysis by Salesman 750
HO 10: Sales Analysis and Profit Analysis by Item 752

HOW TO USE THIS GUIDE

One of the important assumptions concerning this curriculum guide is that it is to be used as a companion publication to a textbook or textbooks. For example, if the instructor wishes to teach all three languages--COBOL, BASIC, and RPG--this would probably mean that three programming language texts would have to be provided the students. Should the instructor decide to teach only two languages, supplemental texts would also be required. References to appropriate textbooks are given in most units.

There are three types of activities provided in this curriculum guide: hands-on programming, paper exercises, and reading or lecture. Usually, the student will have enough knowledge to complete any of these activities from the curriculum guide; however, instructors will probably want to develop supplemental materials of their own to disseminate to students and to use in their lectures. For example, the instructor may make additional transparencies or handouts from the supplementary texts. There are references listed in each unit which will assist the instructor in developing the materials. These items may not be in the instructor's library, but they generally will be available at local college or major city libraries. Instructors are urged to take advantage of these sources to maintain and update their own materials. This maintenance is particularly important as there are major changes taking place in computer technology (i.e., microcomputers) which will affect the relevancy of the material. Also, at the time of this writing, the American National Standards Institute was considering the issuance of a new standard for the COBOL language. It was not possible to include the new standard in this publication; but as computer manufacturers and compiler vendors conform and reissue their COBOL compilers, it will be important for the instructor to stay abreast of these changes.

The preparation of the instructor for this course of study should include formal coursework in business data processing and in the programming languages of COBOL, RPG, and BASIC. Courses of this type are offered at a number of colleges or universities. Although instructors may not be able to attend these courses, other excellent sources of preparation include the courses offered by professional associations. Whenever possible, this curriculum guide will provide references and materials about these in-service activities.

It is expected that the instructor will work through the problem assignments before presenting them to the students. While doing this, the instructor should adapt the material to his or her school's particular computer system. Even COBOL, which is highly standardized, needs at least minor changes for every computer system.

INTRODUCTION

UNIT I

UNIT OBJECTIVES

After studying this unit, the student should be able to begin planning entry into a career in data processing programming.

SPECIFIC OBJECTIVES

After studying this unit, the student should be able to:

1. Match terms and definitions.
2. Recognize true statements relating to the use of this study guide.
3. Relate tasks in the systems life cycle to different DP job types.
4. Differentiate among the various DP job types.
5. List items to prepare for a good employment interview.
6. List the major DP professional organizations and professional magazines.

INTRODUCTION

UNIT I

SUGGESTED ACTIVITIES

I. Instructor

A. Make transparencies; prepare lecture using this curriculum guide and supplemental materials.

B. Provide students with objectives; discuss objectives.

C. Make parallel assignments for reading and programming from the text(s).

D. Provide students with information and assignment sheets; discuss these sheets.

E. Provide students with additional supplementary information and exercises.

F. Give an examination.

II. Student

A. Read objectives.

B. Study information sheets and text as assigned by instructor.

C. Complete assignment sheets and supplementary exercises.

D. Complete examination.

INSTRUCTIONAL MATERIALS

I. Unit content

A. Objectives

B. Notes to the instructor

C. Information sheets

D. Handout 1: Example of Experience-Oriented Resume

E. Transparency masters

1. TM 1: Overview: The Programming Process
2. TM 2: Organization of a Typical DP Department
3. TM 3: DP Job Types

    4. TM 4: Major Programming Job Markets

F. Assignment sheets

    1. AS 1: Visit a Computer Installation
    2. AS 2: Mock Interview
    3. AS 3: Prepare Resume

G. Examination

H. Answers to examination

INTRODUCTION

UNIT I

NOTES TO THE INSTRUCTOR

The background of the student assumed in this manual is 1) the completion of the first data processing study guide, 2) the completion of about one half of a typical COBOL text, such as Shelly and Cashman, and 3) the completion of an introductory general data processing text, such as Shelly and Cashman. This means that students without this background will generally need remedial attention as novice students and will not be able to perform at the level expected in this guide.

COBOL has been selected as the main language of this volume because of its wide spread usage in business data processing. Over ninety percent of business applications on large mainframes are programmed in COBOL. This means that students entering data processing careers who have a good understanding of programming in COBOL will have an advantage over those who do not. This advantage is particularly significant if the student does not intend to graduate from a four-year college program because knowledgeable programmers are in sufficiently short supply that a college degree is sometimes irrelevant.

The availability of a COBOL compiler is particularly important for the student in order to develop good marketable skills in COBOL programming. Although the problems and assignments of this volume will generally be such that they can be done without a COBOL compiler, the student will face a significant disadvantage in the job market without hands-on skills. These hands-on skills can be provided in a variety of ways:

1. On a mainframe computer - The IBM series of 43's, 30's, and the plug compatible competitors are the prototypical machines. Realizing that proper equipment experience is necessary for entry into professional programming positions, this volume will assume that students have access to equipment typical of large corporate environment necessary for professional programming. Later sections of this volume deal with programming in smaller organizational settings but probably over seventy-five percent of programming positions assume this large-scale environment. Since it may be a problem for schools to acquire such expensive equipment, it is suggested that schools enter into a cooperative agreement either with businesses, colleges, or universities to secure this experience for their students. Although not as optimal for student job marketability, minicomputers and microcomputers can also sometimes be used to provide simulated experience for much lower cost.

2. Minicomputers - Some minicomputers have very excellent COBOL compilers. For example, the NCR 3XXX series has an outstanding compiler. Other minicomputers such as the IBM System 34 and 38 also have COBOL capability as do the Hewlett-Packard 3000 series minicomputers and many others. The school with this equipment installed will be able to provide COBOL capability for its students at the cost of obtaining the compiler from the manufacturer or third party software house. Although this equipment will not provide direct hands-on experience with large corporate mainframes, it is a close second in desirability and students can obtain the large system knowledge by special attention to readings and references.

3. Microcomputers - schools with microcomputers installed should consider obtaining a COBOL compiler for these machines. Although COBOL is somewhat of a rarity on microcomputers, it is available for Radio Shack and IBM computers and probably will be increasingly available on the newer sixteen-bit computers. In most cases a school will have to obtain the compiler from a third party software house as few microcomputer manufacturers directly supply this compiler. The magazine BYTE is an excellent source of software information and is available on many newstands. Although using a microcomputer will be one of the lowest cost alternatives, it will also least add to a student's credentials for most corporate programming jobs. However, there are other job opportunities for microcomputer programmers and corporations are also increasingly beginning to use microcomputers as part of their professional EDP system. In summary, although use of a micro to prepare students for a professional data processing programming job has its drawbacks, it is not without some benefits. If the school only has micros, every attempt should be made to broaden exposure of the students to the larger systems more typical of business data processing.

4. No computer facilities - the instructor faces a difficult situation if there are no computer facilities available. If such is the case, this manual should only be attempted with a good deal of caution. There is an attempt in this manual to provide the instructor with sufficient activities to learn most of the concepts included, but this is a weak substitute for actual computer programming. Every attempt should be made to provide at least some exposure to actual programming through cooperative agreements with other schools, local businesses, universities, etc.

INTRODUCTION

UNIT I

INFORMATION SHEETS

I. Terms and definitions

   A. **COBOL** - Common Business Oriented Language

   B. **RPG** - Report Program Generator

   C. **BASIC** - Beginner's All-purpose Symbolic Instruction Code

   D. **Systems Life Cycle** - standard process for developing business computer application

   E. **Systems Analyst** - person that analyzes, designs, and implements business systems

   F. **Applications Programmer** - programmer specializing in developing applications rather than systems software

II. Introduction to Volume II

   A. Purpose of the study guide

   The purpose of this study guide is to assist the student in planning for entry into a data processing career. In particular, it is expected that most students will be interested in becoming business applications programmers; for that reason, this volume concentrates on the three most commonly used business data processing programming languages:

      1. COBOL - Common Business Oriented Language

      2. RPG - Report Program Generator

      3. BASIC - Beginner's All-purpose Symbolic Instruction Code

   B. More units are devoted to COBOL because:

      1. It is generally considered more difficult to learn.

      2. It is the main language on the large computers of large and medium size corporations.

      3. The majority of programming jobs are with large and medium size corporations.

   C. This manual was designed to help the student prepare for jobs in the data processing field, particularly as applications programmers.

   The data processing field is nearly unique in that it is a field in which those without a college degree can still expect not to be barred from entry. Increasingly, employers are expecting prospective employees to be college graduates of data processing or computer science programs. The student with vocational or other precollegiate training in data processing will succeed better in college and in the data processing job market.

So, the best advice seems to be to set your sights high and consider this course of training to be just the beginning of your training which may continue in college and/or on the job.

D. The study guide is organized into four main parts:

  1. Program design, documentation, and system design
  2. Intermediate COBOL programming
  3. BASIC for business data processing applications
  4. Intermediate RPG

III. Overview of the programming process (TM 1)

A. Large corporations have a standard method for taking a new idea for a computer application from start to finish. This is called the systems life cycle.

B. The programming process generally is a subprocess of the systems life cycle. Programming process steps are:

  1. Those performed by systems analyst:
     a. Define objectives of system.
     b. Translate objectives into informational methods.
  2. Those performed by programmer:
     a. Design programs and procedure to produce the required information.
     b. Write, test, and debug programs.
     c. Install programs and systems of programs.
     d. Complete documentation (with systems analysts).

C. Later units give more details on the subprocess performed by the systems analysts in analysis and design.

IV. Data Processing (DP) careers, the DP job markets, and locating a job

A. Organization of the DP department

  1. The organization of the DP department determines the job characteristics of DP personnel. (TM 2)

2. A DP department is divided into three main divisions.
   a. Operations
   b. Development
   c. Maintenance
3. Systems analysts and applications programmers mainly develop new systems or change existing systems. The other personnel mainly operate; i.e., run the computer and associated software. (TM 3)

B. DP careers and job types

1. Data entry operator - job characteristics:
   a. Job is very much like that of typist.
   b. Pay and working conditions similar to that of a typist
   c. The job consists of copying data from source documents, such as pay time cards, sales tickets, etc., into the computer via a CRT terminal or keypunch machine.
2. Computer operator - job characteristics:
   a. Job is similar to other clerical positions, such as a bookkeeper, but has some characteristics of production work.
   b. Pay and working conditions vary with size of computer and the amount of training of an operator.
   c. The job involves following the instructions in a run manual (usually prepared by a programmer) to obtain the necessary files from a long and detailed series of instructions. Pay is lower than that of a programmer.
3. Minicomputer-oriented jobs - job characteristics:
   a. Most small companies that use minicomputers do not have a professional staff of data processors.
   b. Many bookkeepers are retrained to operate the computer, prepare input documents, enter data, and distribute outputs.
   c. Individuals with DP training may be data processing managers/programmers (usually in RPG).
   d. The number of entry-level openings for these jobs is not large because generally the companies tend to develop their existing accounting or other staff personnel to grow into these positions or they purchase software.
4. Applications programmer - job characteristics:

   a. Job involves developing programs to implement designs for business applications systems, such as accounts receivable, inventory, payroll, etc.

   b. Programmers frequently work in teams and on projects.

   c. Pay is higher than preceding jobs; demand is high.

   d. Most design specifications are prepared by those in senior programming positions, such as the programming manager, programmer analyst or systems analyst. Thus, the job is sometimes narrowly defined to include only coding, but newer structured approaches imply more job responsibility with more career potential.

5. Programmer Analyst - job characteristics:

   a. An intermediate position between programmer and systems analyst

   b. There are more job responsibilities for design and user contact than for that of the programmer position, but the job title is often misused. Sometimes there is relatively little difference between this job and that of a programmer.

6. Systems Analyst - job characteristics:

   a. Main responsibility is to be in contact with users to define their problems and develop solutions to them. These solutions may involve anything from accounting to logistics, so the analyst normally has a substantial business background with a college degree(s) and a great deal of experience with that corporation.

   b. Some programmers eventually acquire the skills to become systems analysts even though they do not have the formal background that this job normally requires.

   c. Position has high pay and responsibilities.

7. DP Manager - job characteristics:

   a. Job includes planning and coordinating the activities of the DP department and usually the DP Manager has as direct subordinates a programming manager, systems analysts manager, and operations manager (supervisor of the computer operators and data entry personnel).

   b. Interacts with middle and top management to define priorities for DP and interprets policy and goals to subordinates

   c. Has general management responsibilities for the DP department, so this position requires a substantial DP background

   d. Often holds the position of vice-president

C. Implications of job types for career planning

1. Data entry and operator not particularly attractive; demand is not high, supply is adequate, and pay is low.

2. Minicomputer operator position is attractive, but actual job market is small because hiring is usually done from within the firm. This position is growing especially for BASIC programmers on the smaller minis and micros. Presently, BASIC is the most widely known language.

3. Of the programmer positions, applications programmer is the most attractive in terms of number of jobs. The job potential is good because beginners would not qualify for systems analyst or management positions and would not have the technical background to qualify as a systems programmer.

4. The qualifications for applications programmer in the large and medium size company mainly involve the ability to program in COBOL and RPG. Therein lies the importance of COBOL and RPG to entry-level jobs in the computer field.

5. As further evidence of this importance, the Data Processing Management Association, a professional association of thirty thousand data processing personnel, has put together a curriculum for the entry-level data processing person. That curriculum has a very heavy COBOL emphasis, with as many as five COBOL courses recommended for the beginning level programmer.

D. Major job markets for programming jobs

1. Missouri is in a central location for many programming jobs, although several jobs are located out of state. (TM 4)

2. Most jobs are located in the metropolitan areas.

3. Within the state of Missouri, jobs are advertised in Kansas City and St. Louis newspapers and in national professional magazines. Some jobs are advertised in local newspapers. In Mid-Missouri, the Columbia and Jefferson City newspapers often carry programming want ads. (Professional magazines are discussed in the next section.)

V. Selling yourself in the job interview

A. Importance of interviewing

1. A major factor in acquiring a job is the impression you make in a job interview.

2. How to make a good impression - some hints:

a. Dress conservatively. (Consult Dress for Success by John T. Molloy.)

b. Use positive body language by sitting on the front half of your chair and leaning slightly forward.

c. Maintain good eye contact with the interviewer.

d. Use the latest terminology but be sure you understand the terms and can maintain the conversation.

B. Terminology and job hunting

1. The interviewer will judge your competence by your ability to participate in a meaningful conversation. You should be knowledgeable about the current buzzwords (computer jargon).

2. Study the professional magazines to learn the current buzzwords. (See professional organizations below for professional magazines.)

C. Experience

1. The vicious circle facing the novice is that most employers require you to have job experience, but how can you have experience without having had a job?

2. The solution is to acquire the appearance of experience.

D. The appearance of experience - how to get it

1. Put together a portfolio of all the bigger programs you have written. Make sure that they look professional, are well-documented, are bound in a good quality binder, etc.

2. Put together an experience-oriented resume. For example, don't put down, "I took the course DP 783," but instead describe the course in terms of the experience: "I wrote 1,780 lines of COBOL with the largest program being 500 lines, including Indexed Sequential Files Access, using IBM System 34, etc." In other words, every course should be described as if it were a professional programming assignment. (See HO 1 for an example of a resume which resulted in the selection of a student for a top paying job in San Francisco.)

3. Don't miss a chance to volunteer or get programming experience while you are in school. For example, a student decided to delay graduation so she could program twenty hours a week for the school's administration. This part-time programming experience will greatly help her marketability.

Another student had searched for a job for a year without success. Even though he had a Masters in Business Administration, his confinement to a wheelchair was a handicap. He came back to school and took programming courses, including a project course in which he worked with a realty company designing, writing, and installing an application on the IBM System 34. He was hired by a large meat packing firm to start as a programmer in their Wisconsin office. He said he got the job because of his project with the realty company.

Your school might not have a projects course, but there are many small businesses that need programming help. Don't hesitate to volunteer! For example, lots of people now have microcomputers. Maybe they can't pay, but the experience will greatly help your marketability. Be sure to document your volunteer work to show to your prospective employer during the interview.

E. Applications and contacts

1. Almost all programming shops are chronically short of staff so they are generally looking. Don't be afraid to apply, even though they are not advertising. Send your resume and a brief cover letter.

2. Keep in touch with your classmates, teachers, etc. Employers often depend on their current employees to tell them who is good at a particular school. Hence, let your contacts have a copy of your resume.

F. The single most important thing in job hunting is enthusiasm!

1. If there is any one quality that will land you the job, it's enthusiasm! Be enthusiastic about yourself, your classes, and your computer experiences!

2. Remember, don't be judgmental or lukewarm about yourself or your experiences:

SELL YOURSELF

BY

ENTHUSIASM

VI. Professional organizations and magazines

A. Once you have a job, join a professional DP organization. While in school, join a student chapter.

B. Benefits of a professional organization are:

1. Professional contacts and discussion

2. Job contacts for the future

3. The potential of increasing your professional knowledge by reading, attending classes, etc.

C. Major professional organizations - the three largest:

1. Data Processing Management Association (DPMA)

2. Association for Systems Management (ASM)

3. Association for Computing Machinery (ACM)

D. Notes on professional organizations

1. DPMA is the largest organization. Top computer executives to junior programmers may be members. The organization's publication is entitled Data Management.

2. ASM is oriented to systems analysts and senior programming personnel. Journal of Systems Management is the title of the organization's publication.

3. ACM caters to the scientific programmer and publishes Communications of the ACM.

E. Major periodicals not published by professional organizations

1. Datamation and Computerworld

   a. These publications are expensive but can be found in most city libraries. Most DP shops will circulate copies to their personnel.

   b. Job seekers and novices should study these two periodicals extensively for current terminology and to help them give a good professional representation - the appearance of experience.

4. Computerworld also carries the best national and international job ads.

5. If you're working with small computers, also read:

   a. Byte

   b. PC

HO 1: Example of Experience-Oriented Resume

RESUME

LEE MARKEM, BSBA, CDP

Education

Bachelor of Business Administration in Accounting from the University of Missouri-Columbia; Columbia, Missouri; May 19__; overall GPA 3.12.

Diploma Data Processing from the Cape Girardeau Area Vocational Technical School; Cape Girardeau, Missouri; January 19__; overall GPA 3.5.

Professional Training

Cape Girardeau Area Vocational Technical School - Data Processing Major (Cape Girardeau, Missouri; August 19__ - June 19__ and August 19__ - January 19__). I graduated with a diploma of completion. The program involved three hours of instruction per day, five days per week for eighteen months.

1. RPG II (IBM's Report Generator II), 300 hours of instruction. This section was spread over the second and third semesters of the course. Using the IBM System 32 and the IBM Programming Instruction Guide. Eight business application programs were written along with complete documentation (600 total lines of code).
2. BASIC (IBM), 45 hours of instruction. We used the IBM 360/370 at Southeast Missouri State University to write five short programs (100 total lines of coding).
3. Flowcharting and Documentation Standards, 140 hours of instruction. Twelve weeks were spent on these topics. The class was given large pseudocoded programs to flowchart and document. Documentation was done on all files and printed output. Detailed flowcharts were prepared on all programs.

University Coursework

Computer and Information Systems program (all at University of Missouri-Columbia).

Computer Science 104, Grade - A. Studied basic techniques of programming using the WATFIV compiler. There were six assigned programs and a project required for the class. My project dealt with paying a salary to a group of traveling salesmen based on the amount of products sold in two different geographic areas with two different rates of pay in each division. There were about 400 lines of coding involved in the entire class (and about 225 lines in the most complex program).

Computer Science 203, Grade - B. This course involved instruction in PL/I Checkout, PL/I Optimizing, and PL/C (Incore PL/I from Cornell University) compilers. Instruction on structure, stacks, pointers, and PL/I subroutines and functions. There were six assigned programs with about 550 lines of code.

Accounting 258, Grade - A. COBOL and data processing techniques were covered. Material was taught by coding and flowcharting each program. Input and output layouts were also required on each of these assignments. There were also three assigned programs with about 400 lines of code (and about 145 lines in the most complex program).

Accounting 268, in progress. Intermediate COBOL and accounting information systems. COBOL involved three programs--a complex, sequential file update (450 lines); a simple sort application (100 lines); and an Indexed Sequential File Update (250 lines). COBOL instruction emphasized structured design, pseudocode, hierarchy charts, etc. Accounting Information Systems involved EDP control analysis, systems analysis and design, and management advisory service studied through cases, flowcharting, and discussion.

Professional Experience:

University of Missouri-Columbia; Columbia, Missouri; School of Accounting; Teaching Assistant; August 19__ through present.

1. Developed course material and designed application program for Accounting 448; MAS and Minicomputers in Small Business.
2. Lab instructor in BASIC on the IBM 5100 - assisted graduate students in debugging programs (Winter 19__).
3. Developed and wrote sample application system, simple payroll on the IBM 5100 of Columbia Property Management to illustrate floppy disk and programming concepts.

Supervisor: Dr. Chuck Litecky, University of Missouri-Columbia

Unified Systems, Incorporated; Creve Coeur, Missouri; May 19__ through August 19__.

1. HP 3000 BASIC Programmer (full-time), responsibilities:
    a. Developed and installed cost variance system for Nationwide Financial Services (three thousand lines of coding).
    b. Wrote full documentation on these cost routines and others developed by project team (300 pages).
2. IBM System 370/165 DOS/MVT Operator (nights and part-time).

Supervisor: Mr. Dan Searcy, U.S.I. President

University of Missouri-Columbia; Columbia, Missouri; Campus Computing Center; August 19__ through May 19__ and August 19__ through May 19__.

1. Remote Computer Operator, IBM System 360 Model 20 and Digital Terminals.
2. Assisted students with debugging FORTRAN and PL/I.

Supervisor: Ms. Ruth Pollard

TM 1

Overview: The Programming Process

STEPS:

performed by systems analyst

1. Define objectives of system by interacting with management advisers
2. Translate objectives into informational methods

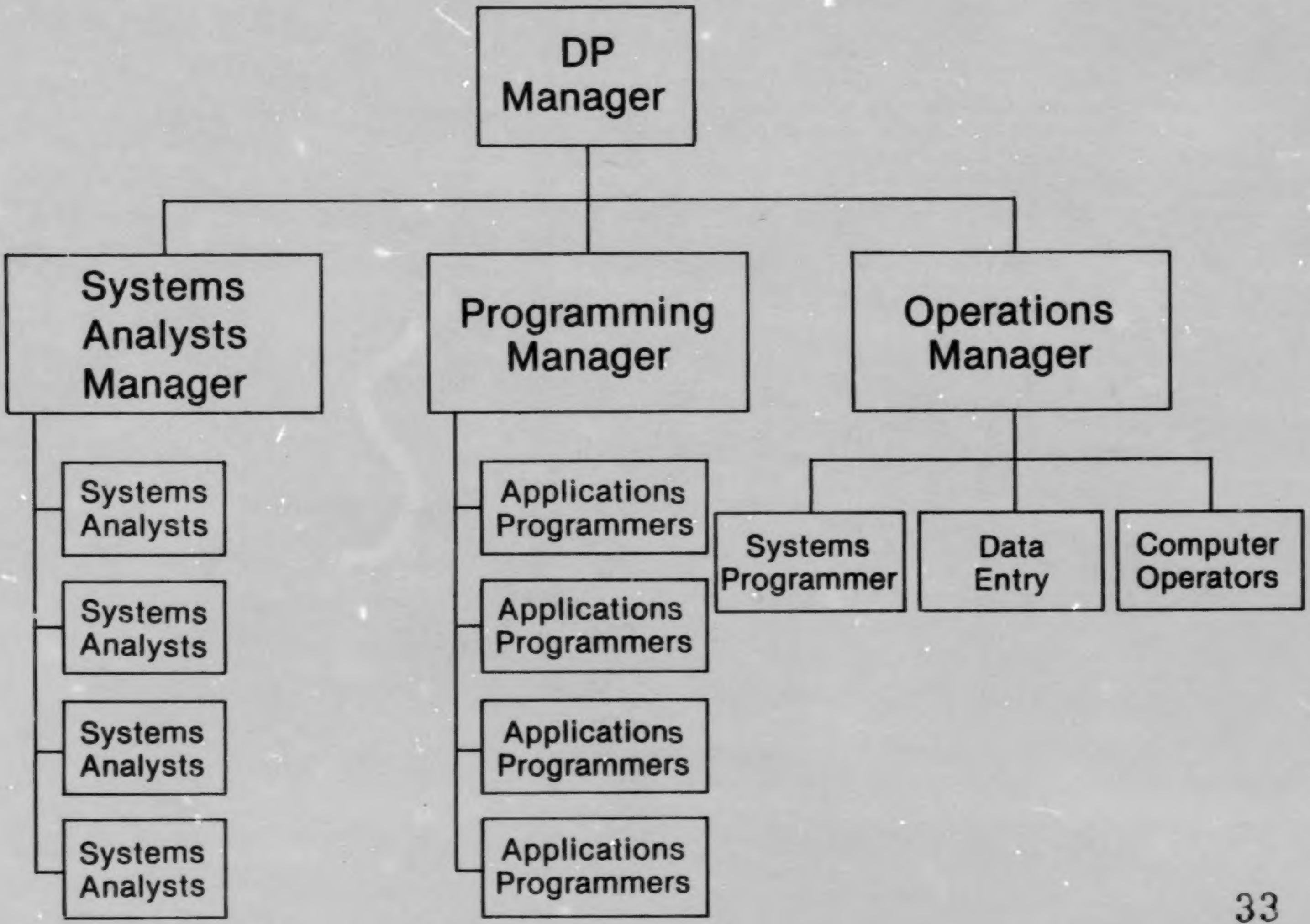
performed by programmer

3. Design programs and procedures to produce the required information
4. Write, test, and debug programs
5. Install programs and systems
6. Complete documentation and perform maintenance as required (with assistance from systems analyst)

performed by systems analyst or auditor

7. Review system's effectiveness

Organization of A Typical DP Department



- DP Manager
  - Systems Analysts Manager
    - Systems Analysts
    - Systems Analysts
    - Systems Analysts
    - Systems Analysts
  - Programming Manager
    - Applications Programmers
    - Applications Programmers
    - Applications Programmers
    - Applications Programmers
  - Operations Manager
    - Systems Programmer
    - Data Entry
    - Computer Operators

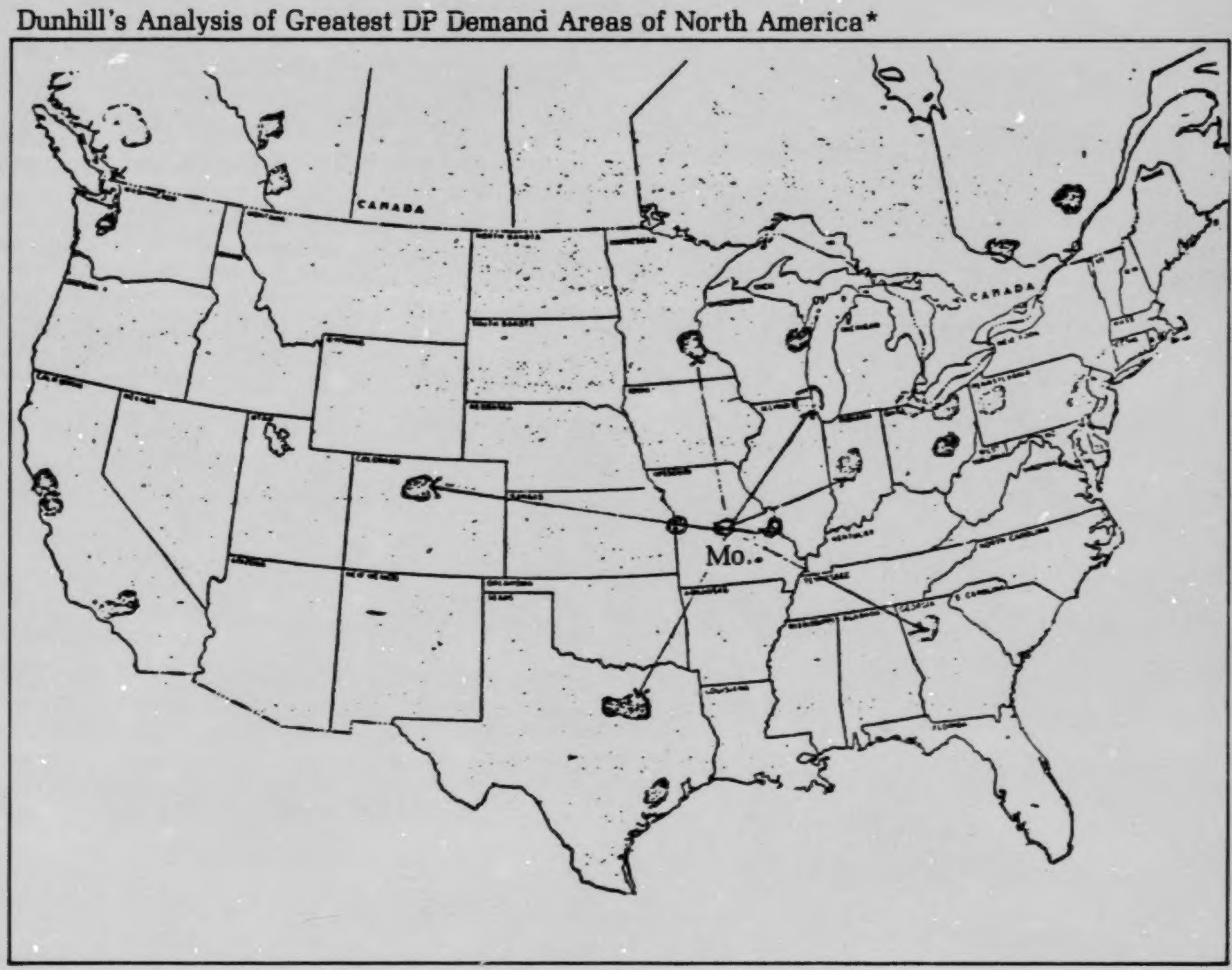
DP Job Types

- Data Entry Operator
- Computer Operator
- Minicomputer-Oriented Jobs
- Programmer
    - Applications
    - Systems
    - Programmer/Analyst
- Systems Analyst
- DP Manager

TM 4

Major Programming Job Markets

Dunhill's Analysis of Greatest DP Demand Areas of North America*


CANADA
CANADA
Mo.

*Source: Adapted from Computer World 12/20/80

ASSIGNMENT SHEET I: Visit a Computer Installation

Either as part of class activities or on your own, visit a typical business computer installation and write a short report about the job duties of the people you observed there.

ASSIGNMENT SHEET 2: Mock Interview

Conduct a mock interview with a classmate. Pick a job want ad from the newspaper and put yourself in the role of the employer interviewing the prospective programmer (your classmate) for the job.

ASSIGNMENT SHEET 3: Revising or Preparing a Resume

Revise or prepare your resume with the career objective of being a programmer. Use the guidelines in Section V of the information sheets and HO I.

INTRODUCTION

UNIT I

EXAMINATION

1. Match each of the definitions in Column A with its correct term in Column B:

		Column A		Column B
____	a.	Analyze and design the job	1.	RPG
____	b.	Implements computer applications	2.	COBOL
____	c.	Standard process for developing systems	3.	Applications programmer
____	d.	Common Business Oriented Language	4.	Systems analyst
____	e.	Report Program Generator	5.	Systems life cycle

2. True-False questions:

____ a. COBOL is generally considered easier to learn than BASIC.

____ b. In the systems life cycle, the programmer generally defines the objectives of the system.

____ c. In the systems life cycle, both the programmer and the systems analyst prepare documentation.

____ d. In the organization of a typical DP department, operations personnel deveiop new computer applications.

3. Contrast the job characteristics of a data entry operator with those of an applications programmer.

4. Which two computer languages are primarily required for programmer jobs in medium and large companies?

5. List two items useful in making a good impression during an employment interview.

6. List the names of two professional organizations and the magazines that they publish.

INTRODUCTION

UNIT I

ANSWERS TO EXAMINATION

1. a. 4
   b. 3
   c. 5
   d. 2
   e. 1

2. a. F
   b. F
   c. T
   d. F

3. Data entry - lower paid, less training required, work more repetitious, consists of keying data

   Applications programmer - works as team member, higher pay, more training required, consists of writing programs

4. COBOL and RPG

5. Conservative clothes, positive body language, eye contact or knowledge of terminology (any two)

6. ASM - Association for Systems Management, Journal of Systems Management
   DPMA - Data Processing Management Association, Data Management
   ACM - Association for Computer Machinery, Communications of the ACM

DOCUMENTATION AND DESIGN TOOLS

UNIT II

UNIT OBJECTIVE

After studying this unit, the student should understand and be able to apply the concepts of major structured and unstructured documentation and design tools.

SPECIFIC OBJECTIVES

After studying this unit, the student should be able to:

1. Match terms and definitions.
2. Distinguish the purpose and usage of Data Flow Diagrams from other tools in documentation and design.
3. Distinguish the purpose and usage of HIPO diagrams from other tools in documentation and design.
4. Distinguish the purpose and usage of hierarchy charts from other tools in documentation and design.
5. Distinguish the purpose and usage of program flowcharts from other tools in documentation and design.
6. Recognize correct statements concerning the use of Warnier-Orr diagrams.

DOCUMENTATION AND DESIGN TOOLS

UNIT II

SUGGESTED ACTIVITIES

I. Instructor

- A. Make transparencies; prepare lectures, using this curriculum guide and supplemental materials.
- B. Provide students with objectives; discuss objectives.
- C. Make parallel assignments for reading and programming from the text(s).
- D. Provide students with information and assignment sheets; discuss these sheets.
- E. Provide students with additional supplementary information and exercises.
- F. Give an examination.

II. Student

- A. Read objectives.
- B. Study information sheets and text as assigned by instructor.
- C. Complete assignment sheets and supplementary exercises.
- D. Complete examination.

INSTRUCTIONAL MATERIALS

I. Unit content

- A. Objectives
- B. Notes to the instructor
- C. Information sheets
- D. Handout 1: Program Flowchart Example
- E. Transparency masters
    - 1. TM 1: Data Flow Diagram Example
    - 2. TM 2: Visual Table of Contents Example
    - 3. TM 3: HIPO Example of an Overview Diagram

        4. TM 4: HIPO Relationships Between Diagrams

        5. TM 5: Most-Used Program Flowcharting Symbols

        6. TM 6: Warnier-Orr Diagram Example

    F. Assignment Sheet 1: Summary and Evaluation of Techniques

    G. Examination

    H. Answers to examination

II. Reference

    A. Welburn, Tyler. "Structured Program Development" in Intro to COBOL. Palo Alto: Mayfield Publishing Co, 1983.

UNIT II

NOTES TO THE INSTRUCTOR

This unit provides background and conceptual content on documentation and design tools. Later units actually develop skills in using the techniques. This forestalls the problem of long, boring assignments for the student and provides a gradual acceleration of documentation difficulty matched to the problem's content.

Unfortunately, no single author provides usage of all the documentation tools. Instructors may wish to require supplementary documentation assignments of their own for later units in this guide.

DOCUMENTATION AND DESIGN TOOLS

UNIT II

INFORMATION SHEETS

I. Terms and functional definitions

   A. **Data flow diagram** - shows logical flow of data

   B. **HIPO** - consists of three types of documents to plan program design

   C. **Visual table of contents** or **Hierarchy charts** - gives concise overview of program structure and hierarchy

   D. **Program flow charts** - provide a graphical picture of program's logic and procedure

   E. **Warnier-Orr diagrams** - use brackets to show logic of processing

II. Data flow diagrams

   A. Conventions

      1. The purpose of the data flow diagram (DFD) is to show only the logical flow of data.

      2. The diagramming technique purposely does not show:

         a. Physical form (tape, disk, etc.)

         b. Resources operating on the data (people, computer, programs, etc.)

         c. Control logic

      3. The DFD is also known as a bubble chart because the only symbols used are round circles.

   B. Example of DFD for a label generating program (TM 1)

III. HIPO - Hierarchy plus Input-Process-Output

   A. Background

      1. HIPO was introduced by IBM Corporation in the early seventies.

      2. It is used for program documentation and design.

      3. HIPO consists of three types of diagrams: visual table of contents, overview diagrams, and detail diagrams.

   B. Objectives of HIPO

1. To provide a structure for understanding the functions of a system of programs

2. To state the functions to be accomplished by each program

3. To provide a visual description of the inputs to be used and the outputs to be produced by each program function

C. Three types of diagrams in HIPO

1. A visual table of contents (VTOC) - much like a hierarchy chart (TM 2)

2. An overview diagram (TM 3)

3. Detail diagrams (TM 4)

D. Evaluation of HIPO

1. Well suited to the design and documentation of structured programs

2. It is useful as a documentation technique because it offers three different levels of detail; e.g., a manager who might need only an overview level.

3. Does not indicate detailed program organization and logic as do program flow charts, so generally must be supplemented with pseudocode

IV. Hierarchy charts

A. Definition

1. Hierarchy charts are graphic structures of each module (COBOL procedure) required by a program, the hierarchical structure of a program, and the relationship of each module to every other module in a program.

2. Hierarchy charts are also called structure charts and/or visual tables of contents (VTOC) (TM 2).

B. Purpose and limitations

1. The purpose is to give a concise overview of program hierarchy and structure. It is useful in the early program design stages to discuss and plan program functions.

2. A hierarchy chart does not show the sequence in which modules will be executed or the conditional decisions that will cause a module to be performed or skipped.

V. Program flowcharts (HO 1; TM 5)

46

A. Definition and purpose

1. Program flowcharts are graphic pictures of a program's logic and procedures and sequence of procedures.

2. They were developed as an early aid to planning and documenting programs.

3. Flowcharts are probably the most used of all the documentation, design, and programming productivity aids; but usage has declined in recent years in favor of other techniques.

B. Evaluation of program flowcharting

1. Expensive and time-consuming to draw

2. Difficult and time-consuming to change

3. Bulky

4. Often cannot be directly translated into structured code; i.e., does not directly support structured code without imposition of rigorous flowcharting standards

5. Still widely used technique; programmers need to know program flowcharting in order to perform maintenance of older programs and their documentation.

VI. Warnier-Orr diagrams

A. Function and background

1. Consist of a series of statements in brackets used with a few other symbols to depict program logic

2. Show both hierarchial relationships and process flows

3. Developed by Jean-Dominique Warnier and supplemented by Kenneth Orr

B. Description of Warnier-Orr Diagrams (TM 6)

1. Process flow is from left to right and top to bottom.

2. Symbol in parentheses below the modules and action descriptions represents the number of times that the module or action is to be executed.

3. A horizontal line drawn above a description indicates a "not" or negated condition.

4. The plus sign in a circle indicates that adjacent modules or actions are mutually exclusive, and only one of a set will be executed.

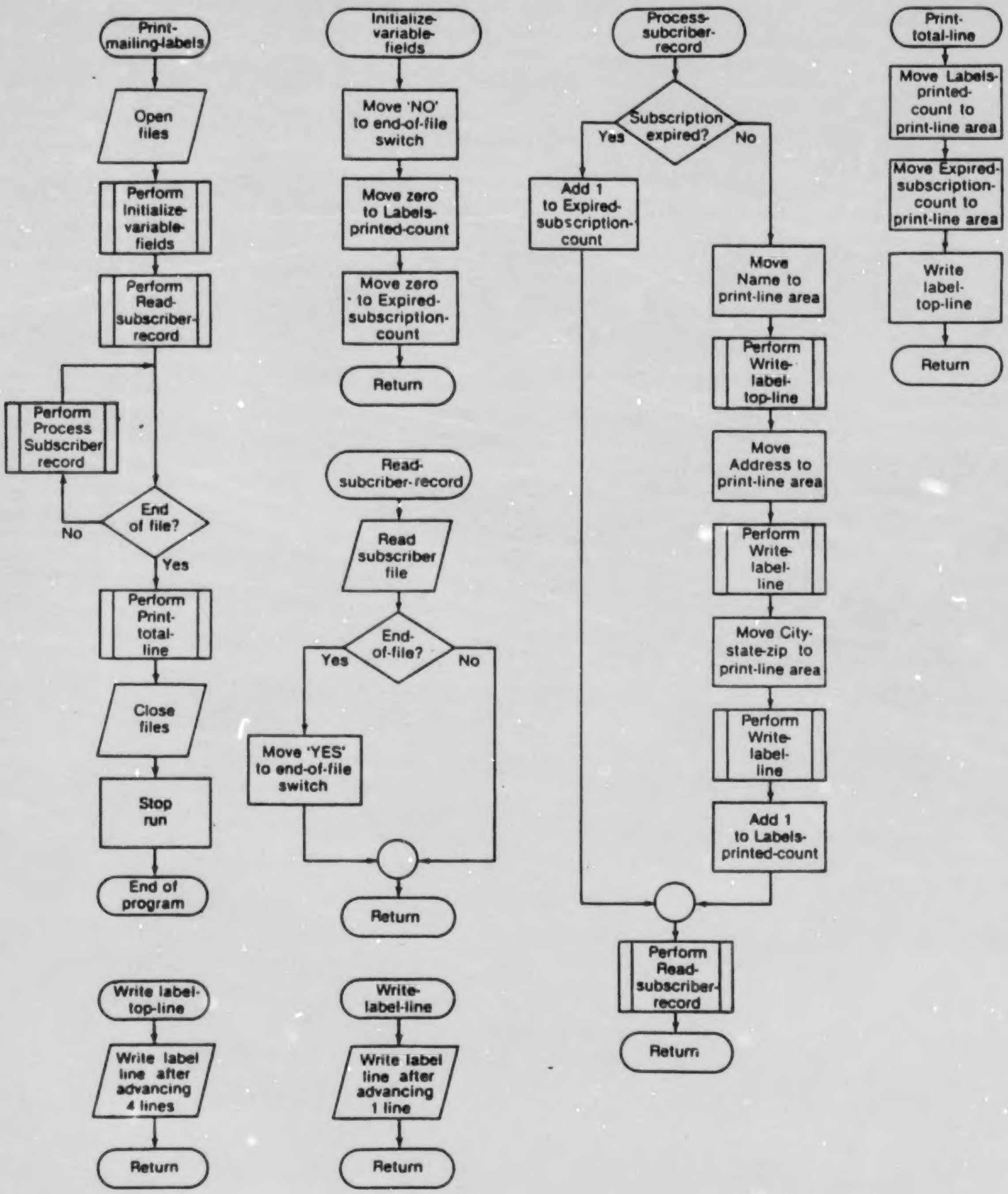
C. Evaluation

1. New and largely untried technique
2. Claimed advantage is that one does not get "hung up" in details of how to get the program written.

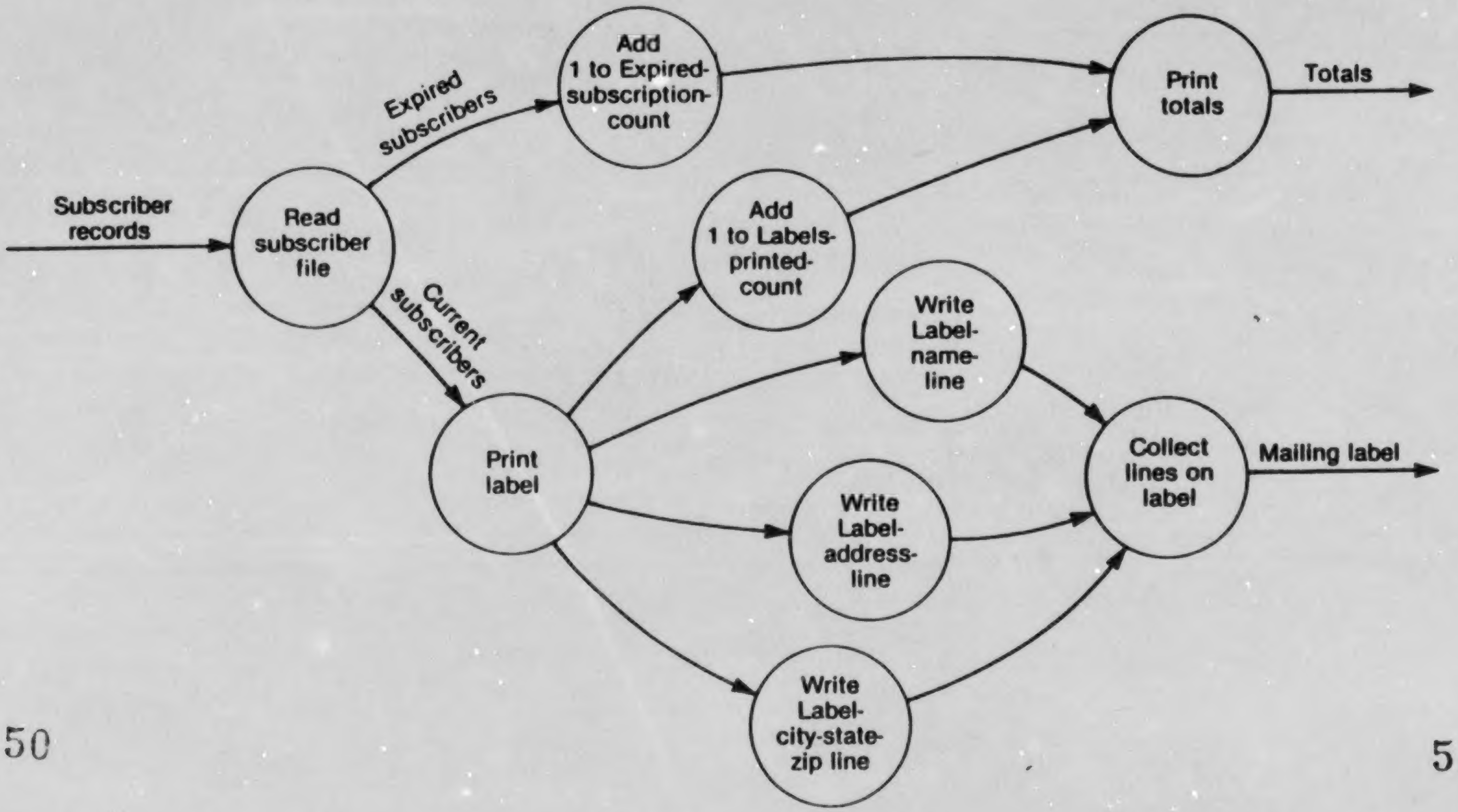
Program Flowchart Example

PROGRAM FLOWCHART

Program ID: LABELS Program name: MAILING LABELS


Print-mailing-labels
Open files
Perform Initialize-variable-fields
Perform Read-subscriber-record
Perform Process Subscriber record
End of file?
No
Yes
Perform Print-total-line
Close files
Stop run
End of program
Initialize-variable-fields
Move 'NO' to end-of-file switch
Move zero to Labels-printed-count
Move zero to Expired-subscription-count
Return
Read-subcriber-record
Read subscriber file
End-of-file?
Yes
No
Move 'YES' to end-of-file switch
Return
Process-subcriber-record
Subscription expired?
Yes
No
Add 1 to Expired-subscription-count
Move Name to print-line area
Perform Write-label-top-line
Move Address to print-line area
Perform Write-label-line
Move City-state-zip to print-line area
Perform Write-label-line
Add 1 to Labels-printed-count
Perform Read-subscriber-record
Return
Print-total-line
Move Labels-printed-count to print-line area
Move Expired-subscription-count to print-line area
Write label-top-line
Return
Write label-top-line
Write label line after advancing 4 lines
Return
Write-label-line
Write label line after advancing 1 line
Return

Data Flow Diagram Example


Subscriber records

Read subscriber file

Expired subscribers

Add 1 to Expired-subscription-count

Current subscribers

Print label

Add 1 to Labels-printed-count

Print totals

Totals

Write Label-name-line

Write Label-address-line

Write Label-city-state-zip line

Collect lines on label

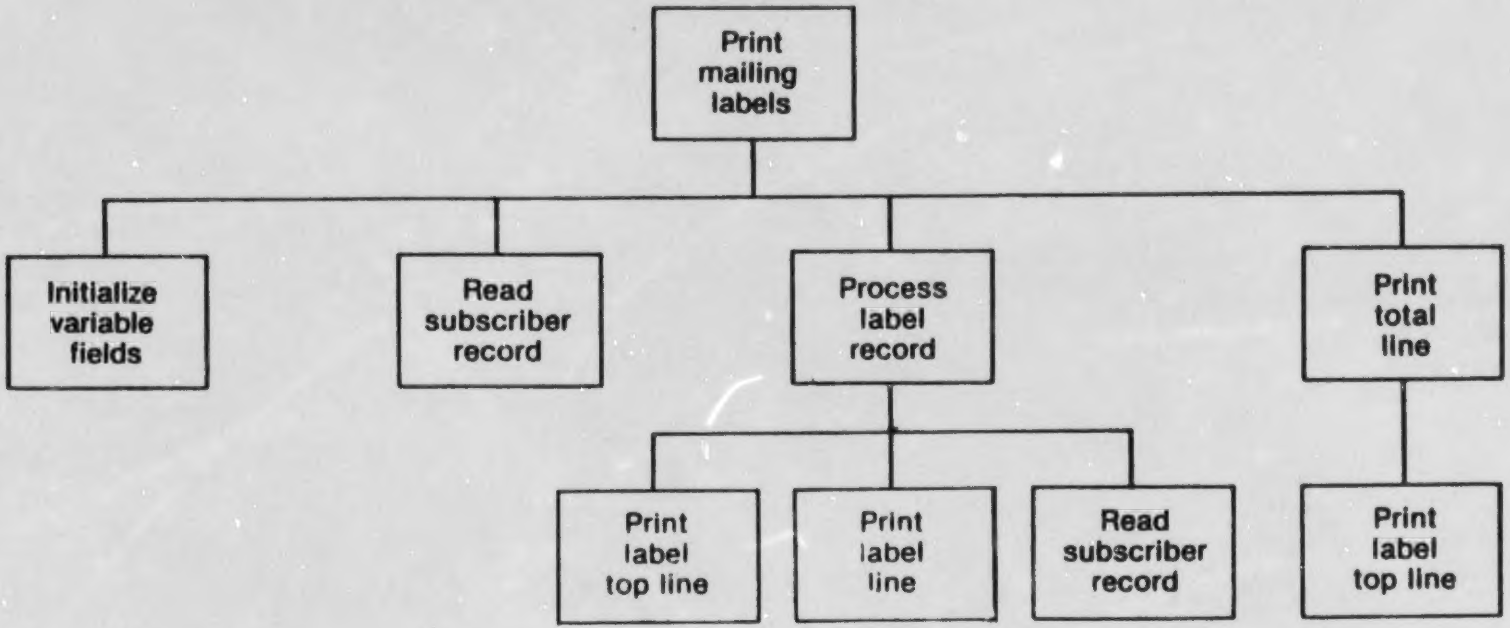
Mailing label

Visual Table of Contents Example

STRUCTURE CHART

Program ID: LABELS

Program name: MAILING LABELS

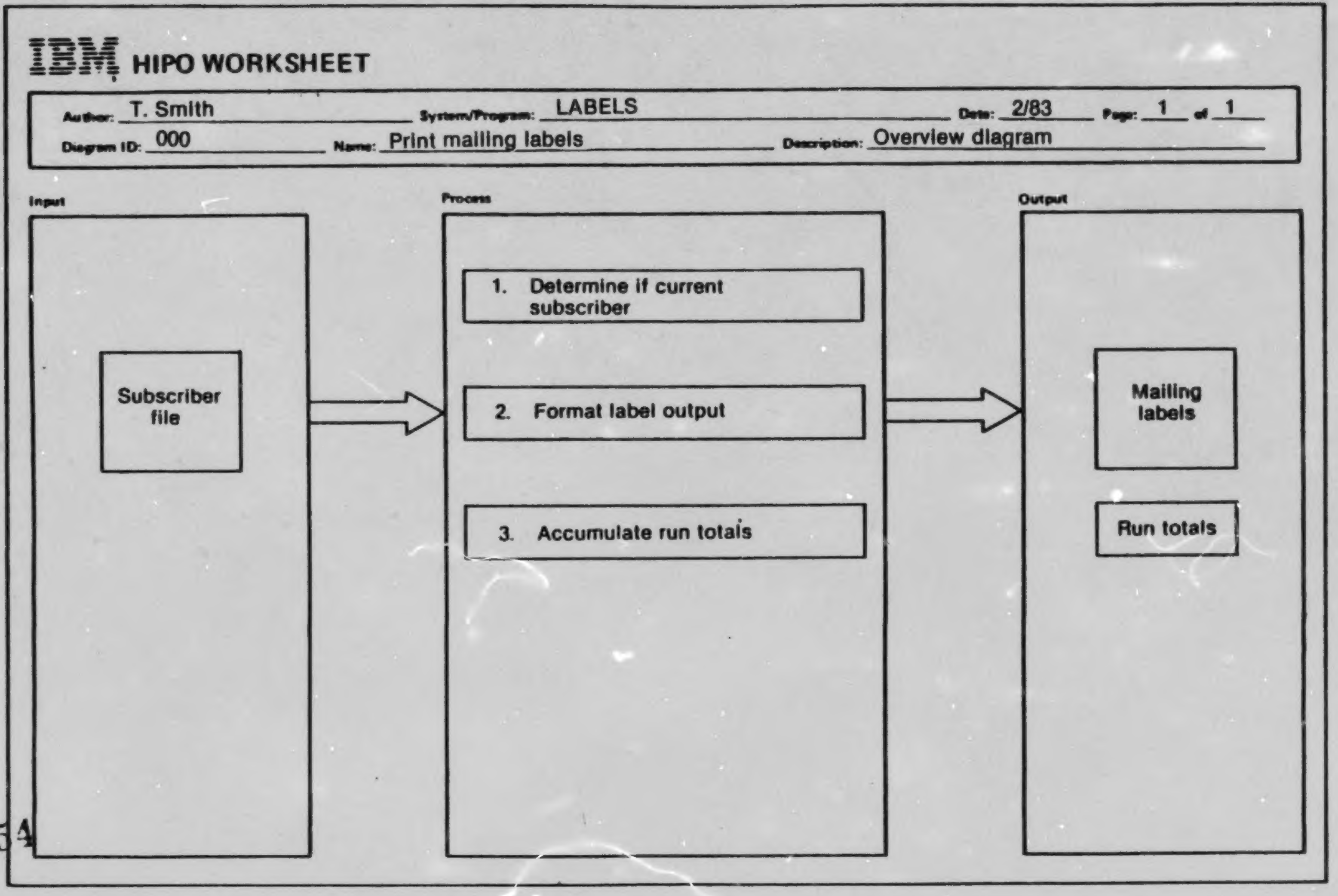

Print mailing labels
Initialize variable fields
Read subscriber record
Process label record
Print total line
Print label top line
Print label line
Read subscriber record
Print label top line

52

53

TM 3

HIPO Example of an Overview Diagram



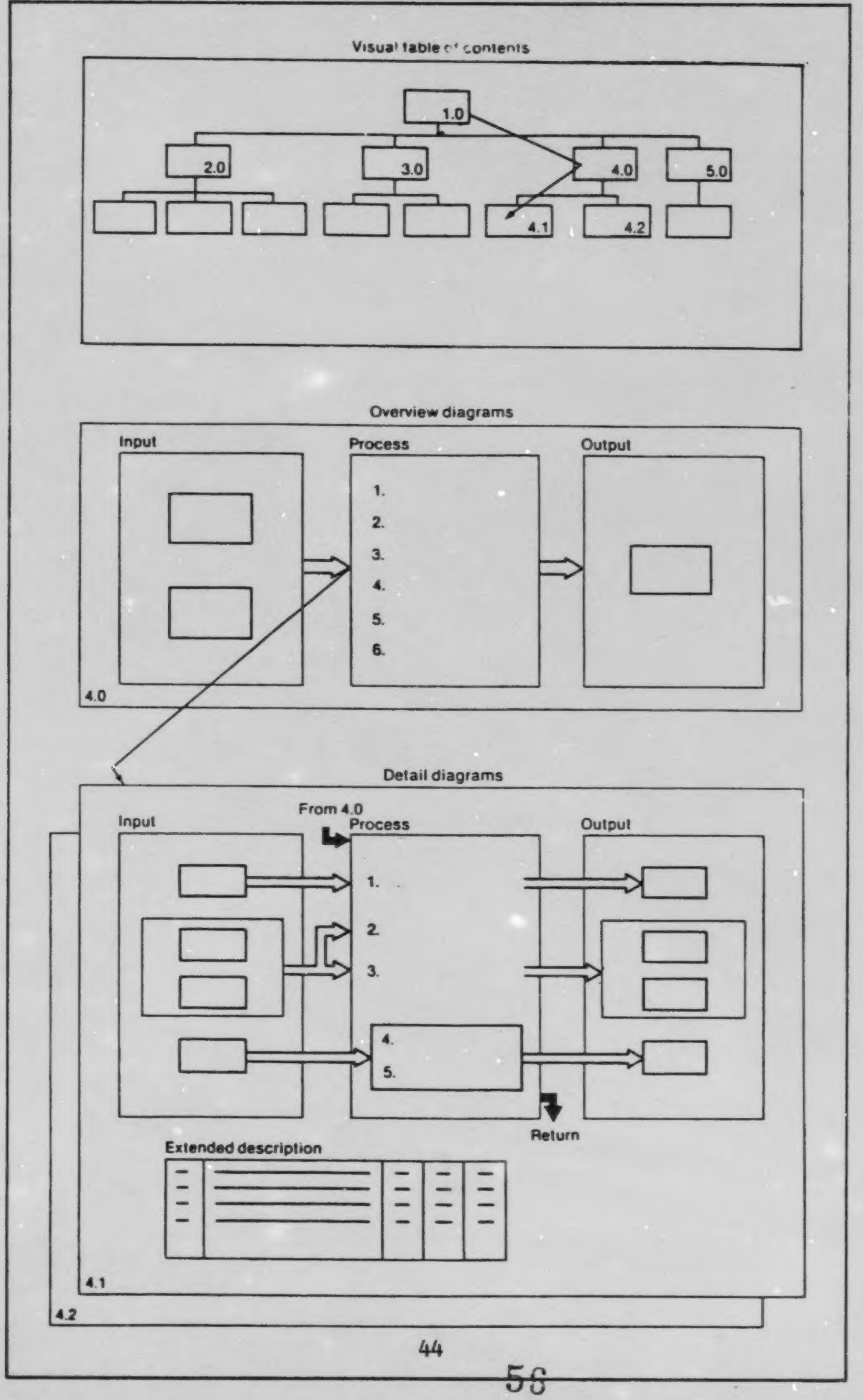
IBM HIPO WORKSHEET

Author: T. Smith
System/Program: LABELS
Date: 2/83
Page: 1 of 1
Diagram ID: 000
Name: Print mailing labels
Description: Overview diagram

Input
Subscriber file
Process
1. Determine if current subscriber
2. Format label output
3. Accumulate run totals
Output
Mailing labels
Run totals

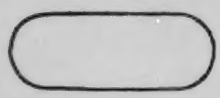
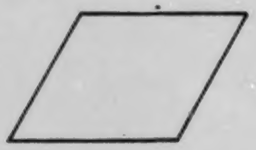
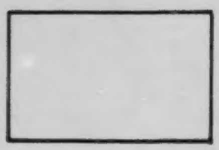
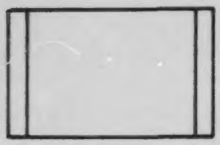
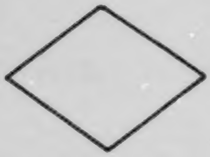
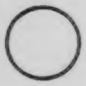
TM 4

HIPO Relationships Between Diagrams


Visual table of contents
1.0
2.0
3.0
4.0
5.0
4.1
4.2
Overview diagrams
Input
Process
Output
1.
2.
3.
4.
5.
6.
4.0
Detail diagrams
From 4.0
Input
Process
Output
1.
2.
3.
4.
5.
Return
Extended description
4.1
4.2

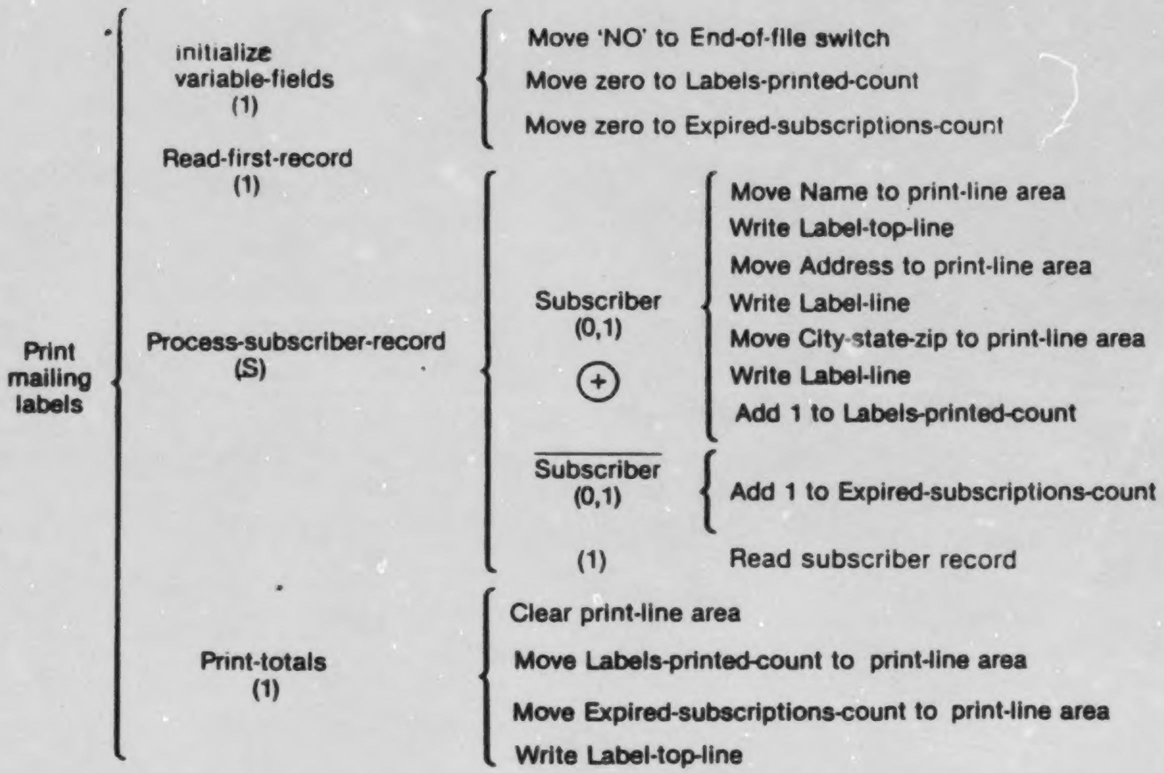
TM 5

Most-Used Program Flowcharting Symbols

Symbol	Name	Usage
	Terminator	to start or stop a program or module
	I/O	any input or output to or from the CPU
	Procedure	any procedure, such as multiply, add, move, etc.
	Predefined process	a predefined process (module) whose details are given elsewhere
	Decision	selection of two or more alternative paths of procedures
	Connector	Connects symbols

TM 6

Warnier-Orr Diagram Example



- Print mailing labels
  - initialize variable-fields (1)
    - Move 'NO' to End-of-file switch
    - Move zero to Labels-printed-count
    - Move zero to Expired-subscriptions-count
  - Read-first-record (1)
  - Process-subscriber-record (S)
    - Subscriber (0,1)
      - Move Name to print-line area
      - Write Label-top-line
      - Move Address to print-line area
      - Write Label-line
      - Move City-state-zip to print-line area
      - Write Label-line
      - Add 1 to Labels-printed-count
    - ⊕
    - $\overline{\text{Subscriber}}$ (0,1)
      - Add 1 to Expired-subscriptions-count
    - (1) Read subscriber record
  - Print-totals (1)
    - Clear print-line area
    - Move Labels-printed-count to print-line area
    - Move Expired-subscriptions-count to print-line area
    - Write Label-top-line

ASSIGNMENT SHEET I: Summary and Evaluation of Techniques

1. Write a short summary of each of the documentation and design techniques. Cover the following items for each technique. Keep it brief.

Name:

Description:

Purpose:

Advantages:

Disadvantages:

Name:

Description:

Purpose:

Advantages:

Disadvantages:

Name:

Description:

Purpose:

Advantages:

Disadvantages:

DOCUMENTATION AND DESIGN TOOLS

UNIT II

ANSWERS TO ASSIGNMENT SHEET

a. Name: DFD (or bubble chart)
Description: network of bubbles (data)
Purpose: shows logical flow of data
Advantage: shows only the logic of the data flow
Disadvantage: does not show control logic

b. Name: HIPO
Description: consists of three documents: VTOC, overview, and detail
Purpose: shows structure or function of systems
Advantage: 1) useful for managers and programmers, 2) shows entire system
Disadvantage: does not show logic and sequence

c. Name: Hierarchy charts
Description: graphic structure of a program and hierarchy
Purpose: shows structure and hierarchy of a program
Advantage: useful for structured overview or in early design stages
Disadvantage: no sequence or decisions

d. Name: Program flowcharts
Description: graphic pictures of program's logic and procedure
Purpose: document programs
Advantage: well known, most used
Disadvantages: bulky, do not directly support structured programming

e. Name: Warnier-Orr diagrams
Description: series of statements in brackets
Purpose: document/design programs
Advantage: presume details of how to write the programs, avoids hangups on how to write
Disadvantages: new, largely untried

DOCUMENTATION AND DESIGN TOOLS

UNIT II

EXAMINATION

1. Match each definition in Column A with its corresponding term in Column B.

	Column A		Column B
____ a.	Shows logical flow of data	1.	Program flow charts
____ b.	Graph of program logic and procedures	2.	Hierarchy charts
____ c.	Has three types of documents	3.	HIPO
____ d.	Shows program hierarchy and structure	4.	Data flow diagram
____ e.	Shows hierarchy and process flows	5.	Warnier-Orr diagram

2. True/False questions:

____ a. The data flow diagram is also known as a bubble chart.

____ b. The data flow diagram does not show control logic.

____ c. The most detailed portion of HIPO documentation is the VTOC.

____ d. HIPO's main purpose is to give a structure for understanding a system of programs.

____ e. A hierarchy chart does not show the sequence in which modules will be executed.

____ f. Program flowcharts are probably still the most widely used documentation technique.

____ g. A Warnier-Orr diagram shows both hierarchy and process flows.

3. Which technique would be intended to document or design an entire system of programs?

4. Which technique predated structured programming?

DOCUMENTATION AND DESIGN TOOLS

UNIT II

ANSWERS TO EXAMINATION

1. a. 4
   b. 1
   c. 3
   d. 2
   e. 5

2. a. T
   b. T
   c. F
   d. T
   e. T
   f. T
   g. T

3. HIPO

4. Programming flowcharts

SYSTEMS ANALYSIS AND DESIGN

UNIT III

UNIT OBJECTIVES

After studying this unit, the student should be able to list the procedures involved in the process of systems analysis and design.

SPECIFIC OBJECTIVES

After studying this unit, the student should be able to:

1. Match terms and definitions associated with systems analysis and design.
2. List reasons for widespread use of computers in business organizations.
3. List three job duties of the systems analyst.
4. Describe each of the phases in the systems life cycle.
5. List activities in the initiation and preliminary investigation phase of the systems life cycle.
6. List fact-gathering techniques used in systems investigation.
7. Describe procedures used in designing system inputs, outputs, controls, processing, and data bases.
8. List items in the detailed programming specifications.
9. Describe methods used to convert applications to new computer systems.

UNIT III

SUGGESTED ACTIVITIES

I. Instructor

- A. Make transparencies; prepare lectures, using this curriculum guide and supplemental materials.
- B. Provide students with objectives; discuss objectives.
- C. Make parallel assignments for reading and programming from the text(s).
- D. Provide students with information and assignment sheets; discuss these sheets.
- E. Provide students with additional supplementary information and exercises.
- F. Give an examination.

II. Student

- A. Read objectives.
- B. Study information sheets and text as assigned by instructor.
- C. Complete assignment sheets and supplementary assignments.
- D. Complete examination.

INSTRUCTIONAL MATERIALS

I. Unit content

- A. Objectives
- B. Information sheets
- C. Handouts
  - 1. HO 1: Example Results of the Preliminary Study
  - 2. HO 2: Example Results of the Detailed Investigation
  - 3. HO 3: Types of Controls
- D. Transparency masters
  - 1. TM 1: DP and Company Organization

        2. TM 2: The Systems Life Cycle
        3. TM 3: Analysis of Collected Facts in Systems Analysis Phase
    E. Assignment Sheets
        1. AS 1: The Preliminary Results Report
        2. AS 2: System Control Design
    F. Answers to assignment sheets
    G. Examination
    H. Answers to examination

II. References

    A. Shelly and Cashman. "Systems Analysis and Design" in Introduction to Computers and Data Processing. Anaheim Publishing Co., 1981.

SYSTEMS ANALYSIS AND DESIGN

UNIT III

INFORMATION SHEETS

I. Terms and functional definitions

   A. **Systems** - a collection of operations and procedures united to accomplish a specific objective

   B. **Systems analysis** - process of decomposing (taking apart) problems into their component parts

   C. **Systems analyst** - job title with responsibilities for analysis, design and implementation

   D. **Organization chart** - form which shows structure and hierarchy of formal chain of command

   E. **Lines of authority** - formal command and reporting lines of management

   F. **Organizational structure** - a corporation's grouping of functions

   G. **Systems life cycle** - process for developing computer applications

   H. **Distributed data processing** - physically and geographically disbursed computing

   I. **Systems Steering Committee** - group responsible for deciding which systems are implemented

   J. **Data Dictionary** - formal definition of an organization's data

   K. **Data Base Administrator** - manager of the data resources of an organization

   L. **GIGO** (Garbage In, Garbage Out) - data doesn't improve in quality just because it came out of a computer

   M. **Subschema** - format and content of the data base within a program/system

   N. **Quality Assurance Group** - those responsible for seeing that new systems are maintainable

   O. **Parallel conversion** - pilot installation of a new system while the old system is still in use

   P. **Direct conversion** - after a certain date only the new system is used

II. The computer in business organizations

   A. Reasons for widespread use of computers:

1. Can reduce costs of doing business
2. Can produce more timely and accurate management information for decision making
3. Can result in intangible benefits, such as improved customer service, leading to increased revenues

B. The computer data processing department

1. Most computer systems in an organization are developed and designed by the data processing department or purchased by them.
2. The persons in the data processing department who are analyzing and designing systems are generally called systems analysts.
3. Systems analysts work with management to assure that outputs from the new system will meet organizational needs.
4. The data processing department manager generally reports to a financial officer of the organization, but increasingly the chief computer executive is himself or herself an officer and reports to the president. (TM 1)
5. In distributed data processing, where systems are geographically disbursed through the organization, data processing department personnel, and especially the systems analysts, are increasingly being relocated in the user departments to provide more direct support.

III. Introduction to systems analysis and design

A. Job duties of the systems analyst are to:

1. Analyze problems and existing business systems
2. Design new systems to solve these problems
3. Implement systems

B. Types of new systems designed by the systems analyst:

1. Conversion of a manual system to one on the computer
2. Conversion of a batch processing system to an interactive system
3. Integration or conversion of a system using separate files to a system using a data base
4. Development of new or advanced systems for provision of management's decision making

IV. Systems life cycle

A. The systems life cycle is a term used to describe a standardized method of conducting systems studies.

B. The systems life cycle is slightly different in many corporations but proceeds through phases. (TM 2)

1. Phase 1 - Initiation and preliminary investigation
2. Phase 2 - Detailed systems investigation and analysis
3. Phase 3 - Systems design
4. Phase 4 - Systems development
5. Phase 5 - Systems installation and evaluation

C. The last stage is depicted as looping back to the beginning to represent:

1. The work is cyclical, proceeding in stages and cycles.
2. The work is never finished but is continuous in nature.
3. Results of the evaluation are used to influence future cycles.

D. An example of a newer approach is the top-down systems analysis or user-done systems analysis, where the work is organized somewhat differently. However, a student should learn the conventional, well-accepted method first.

V. Systems life cycle - Phase I, Initiation and preliminary investigation

A. Initiation

1. Requests for new systems often come from many managers.
2. A Systems Steering Committee will decide which new systems to study and acts as a clearing house and control planning function for longer-range projects.
3. Generally, there will be a backlog of two to three years' work in new projects awaiting initiation.

B. Preliminary investigation

1. Purpose of investigation is to determine if further review of a request or problem is warranted.
2. Often the stated difficulties and the "true" problem are different. The analyst must use considerable personal skills to determine or to distinguish sources of problems from symptoms.
3. An example of a report giving the result of a preliminary study (HO 1)

VI. Systems life cycle - Phase II, Detailed systems investigation and analysis

- A. Detailed systems investigation consists mainly of fact-gathering through:
    - 1. Interviews
    - 2. Questionnaires
    - 3. Collecting documents and forms
    - 4. Observation of procedures
    - 5. Flowcharting of data acquired within the procedures
- B. Analysis of the present system (TM 3)
    - 1. Consists mainly of review and analysis of the collected facts
    - 2. Analysis is partly art but is very rational with only a little science.
    - 3. One analyst calls it the art of asking questions - who? what? where? when? how? and why?
- C. Presentation to management (HO 2)
    - 1. Results of the study to date are presented to management.
    - 2. If management approves, the system will proceed to the next phase. Should it not approve, the system could be and is often stopped here.

VII. Systems life cycle - Phase III, Systems design

- A. Designing the systems output consists of:
    - 1. Defining output requirements
    - 2. Determining types of output media
    - 3. Designing printed reports and terminal-oriented outputs
- B. Designing the systems input consists of:
    - 1. Determining what input data is necessary to generate the required output data given the data in the files
    - 2. Defining the method(s) of data entry into the computer system
    - 3. Determination of batch versus interactive and combinations of the two:
        - a. If batch - design of input records, fields, codes, edit rules, etc.

        - b. If interactive - design of CRT screens, procedures for entering data in user areas, design of related source documents, and special error control

- C. Designing the systems files and data bases
    - 1. When using a data base, the procedure is to construct a data dictionary, which consists of a standardized definition of each data element in the new system. This dictionary must be consistent with the data base.
    - 2. The analyst will consult with the data base administrator when designing access and use of system files. The data base administrator controls the files data and data bases of the organization.
- D. Designing the processing of the system
    - 1. Is accomplished through use of data flow diagrams and systems flow charts
    - 2. At this stage the designer must exercise considerable professional skill, use the knowledge gained in studying the system, and know the organization's line of business.
- E. Designing controls into the system
    - 1. Controls are necessary to:
        - a. Insure data accuracy
        - b. Prevent fraud
    - 2. Summary of types of controls (HO 3)
- F. Systems design considerations
    - 1. System must generally be approved by four levels of management.
        - a. Systems manager
        - b. Data processing manager
        - c. Systems steering committee
        - d. Manager of affected user area
    - 2. Approval is necessary at all these levels because:
        - a. Large sums will be spent in development.
        - b. Jobs or functions of many persons in the various parts of the organization are affected by the new system.

VIII. Systems life cycle - Phase IV, Systems development

A. First step is to develop a detailed project plan specifying the tasks, who is to do each task, the resources necessary, and the individuals (generally a project will have an assigned set of team members) responsible.

B. The analyst(s) then develop detailed programming specifications for each program. Documentation or specifications will include:

1. Brief description of the system
2. System flowcharts
3. Data flow diagrams, HIPO diagrams, Warnier-Orr diagrams, etc.
4. Format for I/O and files
5. Format and content of data base portions used (called subschema)
6. Detailed description of the processing in each program

C. Programmers generally take over main responsibility at this point as the tasks involve:

1. Review of program specifications
2. Design of each program, through the newer, structured techniques of IPO charts, hierarchy charts and pseudocode for each module, or by using the older technique of program flowcharting
3. Coding the programs (most commonly in COBOL)
4. Testing the programs - each program is tested individually, then as a system. The systems analyst may prepare the test specifications or, preferably, a third party, called the Quality Assurance Group, may do the testing.
5. Preparation of documentation for users and operators of the system and for those programmers responsible for maintaining the new system

IX. Systems life cycle - Phase V, Implementation and evaluation

A. Conversion to the new system - two main approaches:

1. Direct conversion - after a certain date, only the new system is used.
2. Parallel conversion - for a period of time (approximately one month) the old and the new systems are run concurrently and checked against each other.

B. Evaluation of the system's effectiveness should be formally conducted by a third party. Internal auditors (accountants, who conduct reviews and are employed by the company) are often used for this purpose.

C. System maintenance - the ongoing correction of old errors and implementation of new changes

1. Increasingly important; over 60-75% of all programming is now done in this area.

2. Good systems design and good life cycle procedures pay off as they enable cost-effective maintenance over the long life (20-30 years) of these systems.

HO 1

Example Results of the Preliminary Study

MEMORANDUM

DATE: March 1
TO: Systems Steering Committee
FROM: Jane George Manager, Systems
SUBJECT: Finding of Preliminary Investigation of Order Entry System

Introduction

A preliminary investigation of the wholesale auto parts order entry system was conducted by the systems department at the request of accounting department supervisor, Lavelle Stinson. The findings of the preliminary investigation are presented in this report.

Objectives of the Preliminary Investigation

The preliminary investigation was conducted to investigate several major complaints concerning the wholesale ordering system. The accounting department has been receiving numerous complaints that orders sent in the mail and given over the telephone have not been shipped promptly. In addition, customers have complained that they have not been notified of parts out of stock for many days after orders have been given to the company. The objective of this preliminary investigation was to determine if the problems actually exist in the form stated, and if so, recommend a course of action.

Finding of the Preliminary Investigation

It is the conclusion of the systems department that the problems as specified do, in fact, exist. With the increased growth of the company during the past several years, the volume of orders for wholesale auto parts has tripled. Orders are currently processed manually by available personnel in the accounting department. The volume is such that many orders are not forwarded to the shipping department for three to four days after receipt. Some customers are cancelling orders because of the delay. It does not appear that any minor revisions in the existing system will provide long-range benefits.

Recommendations

As a result of the preliminary investigation, the systems department recommends that a detailed systems investigation of the order entry system take place. It appears that there is a need to design a new system. It is estimated that one analyst assigned for one month could carry out the investigation recommended at an estimated cost of $3,000.

It is expected at the conclusion of the detailed systems investigation that the analyst will have defined the problem in sufficient depth to be able to offer a specific solution to the order entry problem so that our company may provide the service required by our customers.

George Lacey

Note: The out come of the preliminary investigation should be a report recommending either that no further action should be taken or that a detailed system investigation and analysis should be performed.

HO 2

Example Results of the Detailed Investigation

MEMORANDUM

DATE: April 1
TO: Systems Steering Committee
FROM: Jane George, Manager, Systems
SUBJECT: Detailed Investigation and Analysis of Order Entry System

Introduction

A detailed systems investigation and analysis of the order entry system was conducted as a result of approval given by the management review committee on March 1. The findings of the investigation are presented below.

Objectives of Detailed Investigation and Analysis

The study was conducted to investigate two major complaints of the wholesale auto parts order entry system. Complaints have been received that orders were not being shipped promptly, and customers were not notified of out-of-stock conditions for many days after sending in orders. The objective of this study was to determine where the problems existed and to develop alternative solutions.

Finding of the detailed investigation and analysis

The following problems appear to exist within the order entry system:

1. Three clerks in the accounting department open orders received in the mail and handle all telephone calls relative to orders for wholesale auto parts. The volume of orders is so great during the first three days of the week that orders received are not being forwarded to the shipping department for several days.

2. There is no available means to determine if a part is in stock until the orders are sent to the shipping department. If an item is not in stock, the order is returned to the order department, where the customer is notified. Often there is 5-7 days' delay in notifying custmers of an out-of-stock condition.

Alternative solutions

There are several alternative solutions to improve the order entry system:

1. Revise the manual order entry system. Hire more order entry clerks. Assign specific job duties to accounting personnel. Provide daily inventory reports to order entry clerks to remedy out-of-stock problems. This is only a temporary solution if the company continues to grow.

2. Place the order entry system on the computer. Computer terminals would be installed in the accounting department for order entry clerks. As orders are received, they would be entered into the computer. Orders could be immediately edited for proper part numbers, and checks could be made to determine if stock is available. Estimated costs 1) Systems analysis and design - $17,500; 2) Programming and implementation - $25,000; 3) Training, new forms and maintenance - $5,000; 4) Computer costs 1 year - $12,000.

The systems department recommends alternative 2 as offering the most effective solution to the order entry problem.

Note: The report for the detailed investigation and analysis should contain the results of the investigation and alternative solutions to the problem, together with costs to implement the solutions.

74

HO 3

Types of Controls

Source Document - starting point of good control, prevents GIGO, garbage in, garbage out syndrome, examples are serial numbering, duplicate copies, totaling and balancing of totals.

Input Controls - purpose is to assure complete and accurate conversion (usually by key stroke) of data to machine readable form. Examples: verification, CRT screen control, and editing.

Processing Controls - insures processing is accurate in the programs. Examples: cross checking of files' totals against manual totals; scanning of computer outputs self-check by programs.

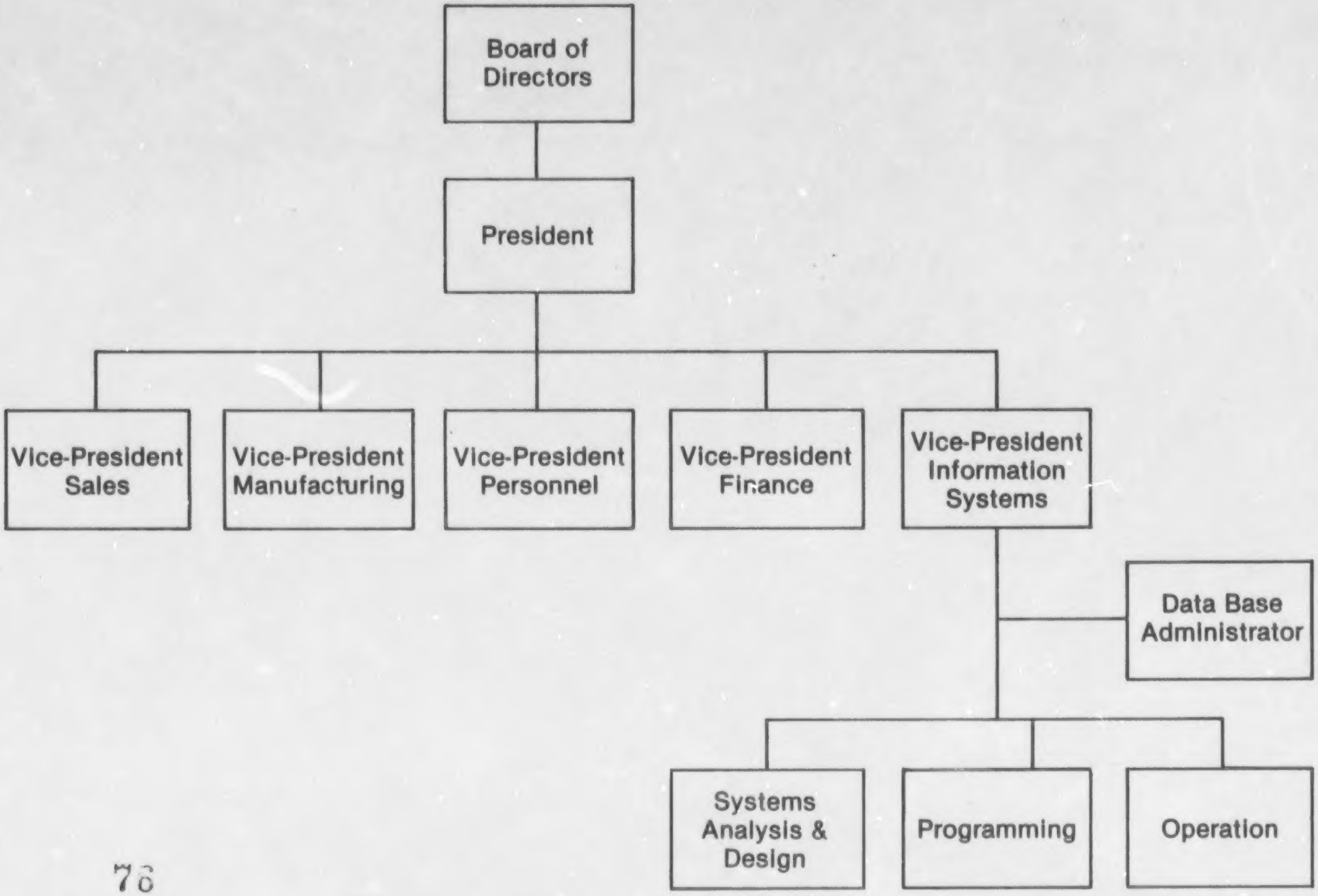
Audit Trail - enables results to be traced back to source or source to be traced forward to final outputs. Examples: logs of transactions, intermediate totals, etc.

Backup Controls - enables files or processing to be recreated when lost or destroyed. Generally, automatic in batch must be designed into interactive systems. Examples: in batch, the grandfather-father-son system. In interactive, on-line logging or recording of all transactions entered from remote terminals.

General Controls - many other types of controls exist--this list could be vastly extended.

TM 1

DP and Company Organization



Board of Directors

President

Vice-President Sales

Vice-President Manufacturing

Vice-President Personnel

Vice-President Finance

Vice-President Information Systems

Data Base Administrator

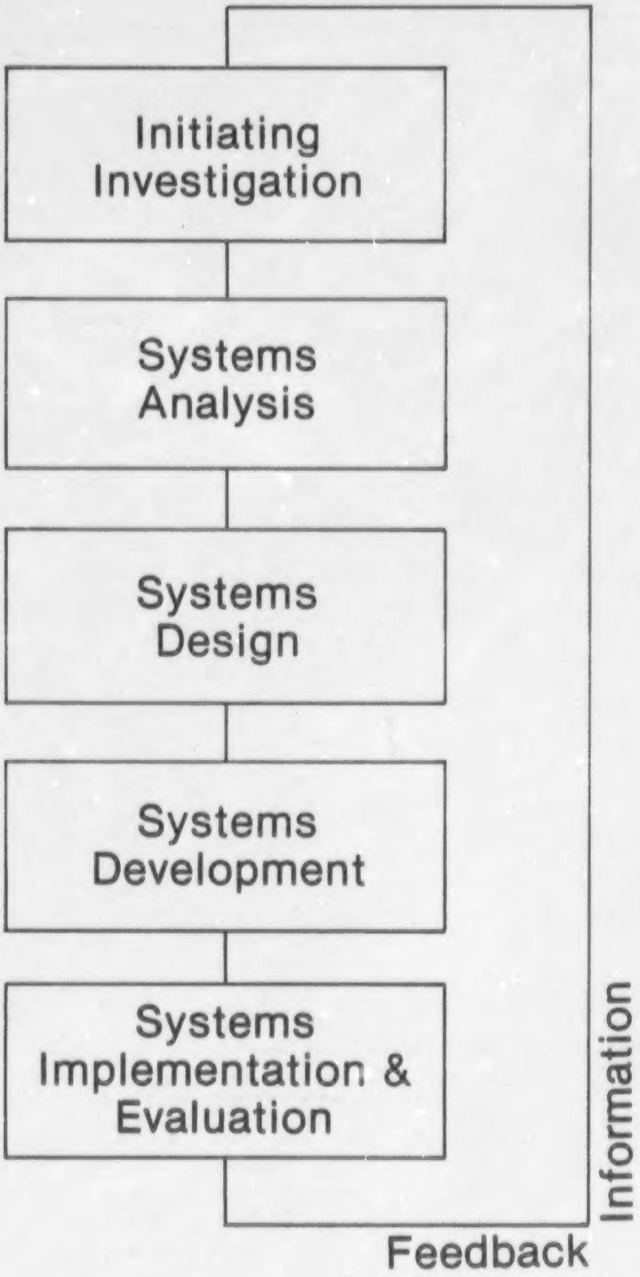
Systems Analysis & Design

Programming

Operation

TM 2

The Systems Life Cycle



Initiating Investigation

Systems Analysis

Systems Design

Systems Development

Systems Implementation & Evaluation

Information Feedback

Analysis of Collected Facts in Systems Analysis Phase

Consists of asking questions:

1. WHO performs the procedure?
2. WHAT procedures are followed?
3. WHERE are the operations performed?
4. WHEN is a procedure performed?
5. HOW is the procedure performed?
6. WHY?

SYSTEMS ANALYSIS AND DESIGN

UNIT III

ASSIGNMENT SHEET I: The Preliminary Results Report

As practice and as an aid to understanding, write a one-paragraph summary of the sample report in HO I.

ASSIGNMENT SHEET 2: System Control Design

Pick two controls from HO 3 and describe each control. Give a specific example of each.

SYSTEMS ANALYSIS AND DESIGN

UNIT III

ANSWERS TO ASSIGNMENTS

Assignment sheet I

Complaints in auto parts entry system were investigated and found to be due to increased volume. A detailed study was recommended at an estimated cost of $3,000.

Assignment sheet 2:

Any two:

Source document - starting point of good control, prevents GIGO syndrome; examples are serial numbering, duplicate copies, totaling and balancing of totals.

Input controls - purpose is to assure complete and accurate conversion (usually by key stroke) of data to machine readable form. Example: verification, CRT screen control, and editing.

Processing controls - insure processing is accurate in the programs. Examples: cross checking of files totals against manual totals, scanning of computer outputs, self-check by programs.

Audit trail - enables results to be traced back to source or source to be traced forward to final outputs. Examples: logs of transactions, intermediate totals, etc.

Backup controls - enable files or processing to be recreated when lost or destroyed. Generally, automatic in batch; must be designed into interactive systems. Examples: in batch, the grandfather-father-son system; in interactive, on-line logging or recording of all transactions entered from remote terminals.

SYSTEMS ANALYSIS AND DESIGN

UNIT III

EXAMINATION

1. Match each of the definitions in Column A with the corresponding term in Column B.

		Column A		Column B
____	a.	Tests new systems	1.	Parallel conversion
____	b.	Pilot installation of new system while old system is still in use	2.	Quality Assurance group
____	c.	Manages data of an organization	3.	Data Base Administrator
____	d.	Decides which systems to implement	4.	Systems Steering Committee
____	e.	Taking problems apart	5.	Systems analysis

2.1. List three reasons for widespread use of computers in business.

a.

b.

c.

2.2. Which department in most corporations is responsible for developing new systems?

3. List three job duties of the systems analyst.

a.

b.

c.

4. Describe the five phases in the systems life cycle.

   a.

   b.

   c.

   d.

   e.

5. List two activities or events in the system initiation and preliminary investigation phase.

   a.

   b.

6. List four fact-gathering techniques used in systems investigation.

   a.

   b.

   c.

   d.

7. Describe the method of designing system inputs.

8. List four items which should be in the detailed programming specifications.

   a.

   b.

   c.

   d.

9. Describe two methods of conversion to the new computer system.

   a.

   b.

SYSTEMS ANALYSIS AND DESIGN

UNIT III

ANSWERS TO EXAMINATION

1. a. 2
   b. 1
   c. 3
   d. 4
   e. 5

2.1. Cost effective, management information, intangible benefits

2.2. Data processing department

3. Analyze, design, and implement system.

4. Phase 1 - Initiation and preliminary investigation
   Phase 2 - Detailed systems investigation
   Phase 3 - Systems design
   Phase 4 - System is developed
   Phase 5 - Install system and evaluate

5. Systems steering committee decides which systems to investigate and requests for new systems come from managers.

6. Any of the following items are acceptable: interviews, questionnaires, document collection, observation, and flowcharting

7. The systems inputs are designed by determining what output data is required and what data is already on file and then by determining the method of processing, batch vs. interactive

8. Specifications include:
   a. Description of the system
   b. System flowcharts
   c. Data flow diagrams
   d. Formats of I/O and files
   e. Descriptions of each program

9. a. Direct conversion - immediate use of the new system
   b. Parallel conversion - old and new systems are run concurrently

PROGRAM DESIGN

UNIT IV

UNIT OBJECTIVE

After studying this unit, the student should be able to list the principal design concepts, recognize correctly designed program modules, and correct poorly designed modules.

SPECIFIC OBJECTIVES

After studying this unit, the student should be able to:

1. Match terms and their definitions.
2. List three reasons supporting the need for good programming design.
3. List three sources of change for existing programs.
4. Identify four general properties of a program module which are used to evaluate the quality of program design.
5. Explain types of cohesion.
6. Explain types of couplings of modules.
7. List rules for determining module size.
8. Explain guidelines for span of control of modules.
9. Correct examples of poorly designed modules.

PROGRAM DESIGN

UNIT IV

SUGGESTED ACTIVITIES

I. Instructor

  A. Make transparencies; prepare lectures, using this curriculum guide and supplemental materials.

  B. Provide students with objectives; discuss objectives.

  C. Make parallel assignments for reading and programming from the text(s).

  D. Provide students with information and assignment sheets; discuss these sheets.

  E. Provide students with additional supplementary information and exercises.

  F. Give examination.

II. Student

  A. Read objectives.

  B. Study information sheets and text as assigned by instructor.

  C. Complete assignment sheets and supplementary exercises.

  D. Complete examination.

INSTRUCTIONAL MATERIALS

I. Unit content

  A. Objectives

  B. Notes to the instructor

  C. Information sheets

  D. Transparency masters

    1. TM 1: Example of a Functionally Cohesive Module

    2. TM 2: Example of a Module with Improper Coincidental Coupling

    3. TM 3: Problems in Common Coupling

    4. TM 4: Control Coupling Example

    5. TM 5: Module Size Rules

  E. Assignment sheets

    1. AS 1: General Review of Program Design

    2. AS 2: Design Exercise

  F. Answers to assignment sheets

  G. Examination

  H. Answers to Examination

II. References.

A. Davis, G. B., M. Olson, and C. R. Litecky. Elementary Structured COBOL Programming. 2nd ed., New York: McGraw-Hill Book Company, in press.

B. Murach, Mike and Paul Noll. "The Principles of Structured Programming" in Structured ANS COBOL, Part 2: An Advanced Course. Fresno, Calif., Murach and Assoc., 1979.

C. Philippakis, A. and Leonard J. Kazmier. Chaps. 1-3 in Advanced COBOL. New York: McGraw-Hill Book Company, 1982.

D. Shelly, Gary B. and Thomas J. Cashman. "Designing and Writing Good Programs" in Advanced Structured COBOL. Brea, CA: Anaheim Publishing Company, 1978.

E. Welburn, Tyler. "Structured Programming" in Advanced Structured COBOL. Palo Alto, CA: Mayfield Publishing Company, 1983.

PROGRAM DESIGN

UNIT IV

NOTES TO THE INSTRUCTOR

Because there is still little uniformity in structured design principles, it is suggested that the instructor obtain and study at least one of the references listed for supplementary background information. The primary reference used for this unit was "Designing and Writing Good Programs" in Advanced Structured COBOL by Shelly and Cashman. Therefore, it is suggested that a different text be selected as a source of supplementary materials.

PROGRAM DESIGN

UNIT IV

INFORMATION SHEET

I. Terms and functional definitions:

   A. **Maintenance** - making changes to existing programs which update their functioning

   B. **Cohesion (or functional cohesion)** - a positive design property of a module involving singleness of purpose or function. All elements within a module should be dedicated to accomplishing one, and only one, purpose.

   C. **Coincidental cohesion** - a negative design property of a module which involves random or multiple-purpose grouped statements - not recommended

   D. **Coupling** - tying modules together through common data or control information - a negative design feature if overused

   E. **Common coupling** - modules use the same data or many modules have access to the same data - a negative design feature if overused

   F. **Control coupling** - modules use the same control information, or control elements are passed between modules - a negative design feature

   G. **Data coupling** - modules pass data to each other - a negative design feature

   H. **Span of control (of a module)** - the number of modules subordinate to a given module

II. Good program design is needed because:

   A. Programs remain in operation for many years.

   B. Programs need maintenance (changes); problems encountered include:

      1. Changes are difficult to make.

      2. Maintenance is expensive; 80% of programming costs are consumed in maintenance.

      3. Maintenance is time consuming.

- C. Good design minimizes operational and maintenance costs. Programs should be designed to be
  - 1. Reliable
  - 2. Easy to modify

III. Reasons for program changes

- A. Changes in governmental or legal requirements
- B. Changes in hardware or software technology
- C. Changes in user needs

IV. Designing good programs

- A. Theory of structured design
  - 1. Based upon work of Stevens, Myers, Constantine, and many others
  - 2. Based upon general properties of program modules which are used to evaluate the quality of program design
    - a. Cohesion
    - b. Coupling
    - c. Module size
    - d. Span of control

V. Cohesion of program modules

- A. Cohesion - all elements within the module are dedicated to accomplishing one, and only one, task within the program
- B. Types of cohesion
  - 1. Functional cohesion (TM I)
    - a. Main type
    - b. Means single purpose
  - 2. Coincidental cohesion
    - a. Opposite of functional
    - b. Means little or no relationship among elements within a module
    - c. Essentially a random collection of elements

91

d. Not a recommended type of cohesion (TM 2)

VI. Coupling of program modules

  A. Coupling - one module communicates with another module or is tied to another module through dependencies introduced by the method of communication or control

  B. General rules

    1. Modules should be coupled only by each module being entered at its beginning and exiting at its end.

    2. A module is entered only by another module directly above it in a hierarchy chart.

  C. Types of coupling

    1. Common coupling - many modules have access to the same data in a program. (TM 3).

      a. Negative type of coupling because one module can change data and negatively affect the other modules' use of the data.

      b. Avoidable through four rules:

        1) Constants in the data division should be used for one and only one purpose.

        2) Any switches should be used for one and only one purpose.

        3) All tables will have at least one index or subscript which is unique to a single table.

        4) In general, any defined data will be used only for one purpose.

    2. Control coupling - involves coupling in which control elements are passed between modules

      a. Negative because one module's changes can affect another.

      b. General rule is to avoid, where possible, passing indicators from module to module; i.e., avoid tightly coupled modules. (TM 4)

    3. Data coupling - one module passes data to another.

      a. General rule - a module should have access only to that data necessary for its function.

VII. Size of program modules (TM 5)

A. General rule - a module should be no larger or no more complex than can be understood in one sitting.

B. A module should be changeable in no more than one-half hour

C. Maximum size - generally less than 50 lines

D. Minimum size - no less than 7-8 lines

VIII. Span of program module control - the number of modules which are subordinate to a module

A. General rule - span of control should be from two to seven modules.

B. Sometimes there are too few functions or modules. When this occurs, the modules might be absorbed into the next higher level module.

IX. Summary of program design

A. In good design:

1. Modules are cohesive and have unity of purpose.

2. Modules are not coupled, or coupling is minimized.

B. Good design takes judgment and forethought.

C. The most general guideline is to make each module easy to modify and maintain.

d. Not a recommended type of cohesion (TM 2)

VI. Coupling of program modules

- A. Coupling - one module communicates with another module or is tied to another module through dependencies introduced by the method of communication or control
- B. General rules
  - 1. Modules should be coupled only by each module being entered at its beginning and exiting at its end.
  - 2. A module is entered only by another module directly above it in a hierarchy chart.
- C. Types of coupling
  - 1. Common coupling - many modules have access to the same data in a program (TM 3).
    - a. Negative type of coupling because one module can change data and negatively affect the other modules' use of the data.
    - b. Avoidable through four rules:
      - 1) Constants in the data division should be used for one and only one purpose.
      - 2) Any switches should be used for one and only one purpose.
      - 3) All tables will have at least one index or subscript which is unique to a single table.
      - 4) In general, any defined data will be used only for one purpose.
  - 2. Control coupling - involves coupling in which control elements are passed between modules
    - a. Negative because one module's changes can affect another.
    - b. General rule is to avoid, where possible, passing indicators from module to module; i.e., avoid tightly coupled modules. (TM 4)
  - 3. Data coupling - one module passes data to another.
    - a. General rule - a module should have access only to that data necessary for its function.

VII. Size of program modules (TM 5)

    A. General rule - a module should be no larger or no more complex than can be understood in one sitting.

    B. A module should be changeable in no more than one-half hour

    C. Maximum size - generally less than 50 lines

    D. Minimum size - no less than 7-8 lines

VIII. Span of program module control - the number of modules which are subordinate to a module

    A. General rule - span of control should be from two to seven modules.

    B. Sometimes there are too few functions or modules. When this occurs, the modules might be absorbed into the next higher level module.

IX. Summary of program design

    A. In good design:

        1. Modules are cohesive and have unity of purpose.

        2. Modules are not coupled, or coupling is minimized.

    B. Good design takes judgment and forethought.

    C. The most general guideline is to make each module easy to modify and maintain.

Example of a Functionally Cohesive Module

```
C011-SUMMARIZE-TOTAL-PAY.
        COMPUTE REPORT-TOTAL-GROSS-PAY =
                REPORT-TOTAL-GROSS-PAY  +
                TRANSACTION-GROSS-PAY.
        COMPUTE REPORT-TOTAL-NET-PAY =
                REPORT-TOTAL-NET-PAY  +
                TRANSACTION-NET-PAY.
```

Example of a Module with Improper Coincidental Coupling

MODULE-C.

Compute net-pay

Determine if new page needed

If Amt-owned negative do error-routine

etc.

TM 3

Problems in Common Coupling

Avoidable through four rules:

1. Constants used for only one purpose.
2. Switches used for only one purpose.
3. Tables have unique subscripts.
4. Any defined data will be used for one and only one purpose.

Control Coupling Example

```
A010-HI-LO-EQUAL-CONTROL.
    IF TRANSACTION-ID GREATER THAN
        MASTER-ID MOVE 'Y' TO
        TRAN-GR-MAST.
    .
    .
    .
B020-WRITE-MASTER-CONTROL.
    IF TRAN-GR-MAST NOT EQUAL 'Y'
        PERFORM ERROR-ROUTINE.
```

Here B020- is coupled to A010. Can be avoided by having B020- repeat the original test.

Module Size Rules

1. Understandable in one sitting
2. Changeable in one-half hour
3. Generally less than 50 lines
4. Generally more than 7-8 lines

ASSIGNMENT SHEET 1: General Review of Program Design

1. List three sources of changes in programs. Pick a specific company in your area and identify specific sources for that company.

2. List four general properties of module design which are used to evaluate the quality of program design.

3. Explain the following terms:

   a. Cohesion

   b. Common coupling

   c. Control coupling

   d. Data coupling

ASSIGNMENT SHEET 2: Design Exercises

1. Write a COBOL module which demonstrates coincidental cohesion.

2. Rewrite the modules in TM 4 to remove control coupling(s).

PROGRAM DESIGN

UNIT IV

ANSWERS TO ASSIGNMENT SHEETS

Assignment Sheet 1:

1. Any three:
   - a. Government
   - b. Legal
   - c. User needs
   - d. Program performance

2. Cohesion
   Coupling
   Module size
   Span of control

3. a. Cohesion - unit of purpose
   b. Common coupling - same data accessed by many modules
   c. Control coupling - control elements passed between modules
   d. Data coupling - modules pass data to each other

Assignment Sheet 2:

(NOTE: Almost any COBOL module with multiple nonrelated functions will be satisfactory.)

2.

```
A010-HI-LO-EQUAL-CONTROL.
B020-WRITE-MASTER-CONTROL.
    IF TRANSACTION-ID GREATER
        THAN MASTER-ID
            PERFORM ERROR-ROUTINE.
```

PROGRAM DESIGN
UNIT IV

EXAMINATION

1. Match each of the terms in Column A with its correct functional definition in Column B:

	Column A		Column B
____ a.	Span of control	1.	Negative design feature involving common data
____ b.	Cohesion	2.	Modules pass data
____ c.	Common coupling	3.	Modules pass control information
____ d.	Data coupling	4.	A module has a single purpose
____ e.	Control coupling	5.	A module has multiple purposes
____ f.	Coincidental cohesion	6.	Number of subordinate modules
		7.	Modules use the same data

2. List three reasons for good program design.

a.

b.

c.

3. List three sources of changes in computer programs.

a.

b.

c.

4. Identify four general properties of a program module which are used to evaluate program design quality by placing an X beside the correct word(s).

____ a. Maintenance
____ b. Span of control
____ c. Program costs
____ d. Module size
____ e. Coupling
____ f. Cohesion

5. Explain the term "cohesion" and give an example of good cohesion in a COBOL module.

6. Explain control coupling.

7. List any two rules for determining the size of modules.

8. Give a rule for determining the span of control of modules.

9. Correct the following COBOL module to remove a data coupling.

```
CALCULATE-GROSS-PAY-RTN.
    COMPUTE GROSS-PAY = HOURS-WORKED*PAY-DATE

    PERFORM CALCULATE-NET-PAY.

CALCULATE-NET-PAY.
    COMPUTE NET-PAY = GROSS-PAY - DEDUCTIONS
```

PROGRAM DESIGN

UNIT IV

ANSWERS TO EXAMINATION

1. a. 6
   b. 4
   c. 7
   d. 2
   e. 3
   f. 5

2. Changes are difficult, expensive, time consuming

3. Government
   Legal
   Technology
   User
   Program functioning

4. a.
   b. x
   c.
   d. x
   e. x
   f. x

5. Cohesion - means a module should have a single purpose.

   Example:

```
CALCULATE-DISCOUNT-RTN.
     IF PRODUCT-TYPE = 'A'
          COMPUTE  DISCOUNT  =  SALES-PRICE  *  LOW-DISCOUNT-
          CONSTANT
     ELSE
          MOVE ZERO TO DISCOUNT
```

   (NOTE: Many other examples of cohesion are possible.)

6. Control coupling means that modules pass control information between themselves. Although necessary, if it is used excessively, it can affect maintenance negatively.

7. Any two of the following:

   Modules should be understandable in one sitting.
   Modules should be less than 50 lines.
   Modules should be no less than 7-8 lines.
   Modules should be changeable in one-half hour.

8. Span of control should be from two to seven modules.

9. Several approaches are possible - the main point is to remove the data coupling caused by passing GROSS-PAY; e.g.:

```
CALCULATE-PAY.
     COMPUTE GROSS-PAY = HOURS-WORKED*PAY-RATE

     COMPUTE NET-PAY = GROSS-PAY - DEDUCTIONS.
```

NESTED AND OTHER IF STATEMENTS

UNIT V

UNIT OBJECTIVE

After studying this unit, the student should be able to write nested IF statements and related statements.

SPECIFIC OBJECTIVES

After studying this unit, the student should be able to:

1. Match terms and definitions.
2. Write simple nested IF statements with matched IF and ELSE statements.
3. Write simple nested IF statements with unmatched IF and ELSE statements.
4. Write simple nested IF statements with semi-matched IF and ELSE statements.
5. Write nested IF statements with use of NEXT SENTENCE clauses.
6. Write multiple level nested IF statements.
7. Give rules for and discuss design of nested IF statements.
8. List steps in writing a complex IF statement from specifications.
9. Explain how writing submodules with special nested IF statements simplifies complex conditionals.
10. Write combined (logical) IF statements.
11. Write complex IF statements using parenthesis.

NESTED AND OTHER IF STATEMENTS

UNIT V

SUGGESTED ACTIVITIES

I. Instructor

   A. Make transparencies; prepare lectures, using this curriculum guide and supplemental materials.

   B. Provide students with objectives; discuss objectives.

   C. Make parallel assignments for reading and programming from the text(s).

   D. Provide students with information and assignment sheets; discuss these sheets.

   E. Provide students with additional supplementary information and exercises.

   F. Give examination.

II. Student

   A. Read objectives.

   B. Study information sheets and test assignments.

   C. Complete assignment sheets and supplementary exercises.

   D. Complete examination.

INSTRUCTIONAL MATERIALS

I. Unit content

   A. Objectives

   B. Information sheets

   C. Transparency masters

      1. TM 1: Multiple Level Nested IF's with Inverted Pairing

      2. TM 2: Example Programming Specifications for Writing a Nested IF

      3. TM 3: The Most Inclusive Condition

      4. TM 4: Summary of Steps in Writing Nested IF's

        5. TM 5: Use of Submodule to Simplify Complex IF Statement
        6. TM 6: Example of a Complex IF Simplified Through Parenthesis
    D. Assignment sheets
        1. AS 1: Simple Nested IF's with Matched and Unmatched ELSE's
        2. AS 2: Semi-matched and NEXT SENTENCE IF Sentences
        3. AS 3: Multiple Level IF Sentences
        4. AS 4: Designing Nested IF's
        5. AS 5: Combined and Complex IF Statements
    E. Answers to assignment sheets
    F. Examination
    G. Answers to examination

II. References

A. Davis, G. B., M. Olson, and C. R. Litecky. Elementary Structured COBOL Programming. 2nd ed., New York: McGraw-Hill Book Co., in press.

B. Shelly, Gary B. and Thomas J. Cashman. "Nested IF Statements" in Introduction to Structured COBOL Programming. Brea, CA: Anaheim Publishing Company, 1977.

NESTED AND OTHER IF STATEMENTS

UNIT V

INFORMATION SHEET

I. Terms and definitions

A. **Nested IF statement** - IF statement contained within other IF statement(s), often paired with its alternative statement, the ELSE statement

B. **ELSE statement** - the alternative statement or false alternative action to be taken when an IF statement's main condition is not true

C. **Matching IF/ELSE statements** - IF/ELSE statements with an equal number of IF's and ELSE's, generally used for mutually exclusive conditions

D. **NEXT SENTENCE** - COBOL command to cause processing to proceed to the beginning of the sentence immediately following the current sentence

E. **Most inclusive statement** - in a programming specification, the condition which applies to the most possible actions

F. **Logical operators** - "AND" and "OR" used to specify combined conditional clauses

II. Simple nested IF statements: matched IF and ELSE statements

A. General Format:

```
IF condition

      execute statements following the 'IF condition' if the condition is
      true.

ELSE

      execute statements following the 'ELSE' if the condition is not
      true.
```

B. Example:

```
IF TRANSACTION-CODE = 'ADD'

      PERFORM ADD-ROUTINE

ELSE

      PERFORM ERR-ROUTINE.
```

1. Notes on example:
    a. In the example, if the transaction code is equal to 'ADD', the ADD-ROUTINE is performed; otherwise, the ERR-ROUTINE is performed.
    b. Note the period following the sentence. Do not forget to place this period to end the sentence.
    c. This is the most common format.

III. Simple nested IF statements: Unmatched IF and ELSE statements

A. General Format:

```
IF condition-1

          executed statements-1 if true

          IF condition-2

                    executed statements-2 if true.

          ELSE

                    executed statements-3 if second IF is not true.
```

B. General Notes:

1. Can be numerous IF's; IF's do not have to be paired with ELSE's.
2. If ELSE's are used, they will be paired with IF's.
3. The recommended practice is to write each ELSE in the same column with its paired IF.
4. ELSE's are paired with IF's in order of the ELSE's occurrence; the first ELSE is paired with the last IF; the second ELSE is paired with the second last IF, etc.

C. Example: Simple nested IF's with unmatched IF and ELSE statements

```
IF SALES-TYPE EQUAL '1'

          IF DISCOUNT-CODE = 'A'

                    COMPUTE DISCOUNT = SALES-AMT*.1

          ELSE

                    MOVE ZERO TO DISCOUNT.
```

IV. Simple nested IF statements: semi-matched IF/ELSE statements

A. General Format:

format has many variations but typically:

```
IF condition-1

     statement-1

ELSE

     IF condition-2

          statement-2

     ELSE

          IF condition-3

               statement-3.
```

B. General notes on this format

1. If condition-1 is true, only statement-1 is processed in this sentence.
2. If condition-1 is not true, then condition-2 is tested; IF condition-2 is true, then statement(s)-2 are processed.
3. The remaining possibility is that both condition-1 and condition-2 are not true, and then condition-3 is tested; IF true, statement-3 is done.

C. Examples of nested IF with semi-matched IF/ELSE statements:

```
IF TYPE-INPUT = 'A'

     MOVE ADD-CONSTANT TO TYPE-OUTPUT

ELSE

     IF TYPE-INPUT = 'D'

          MOVE DELETE-CONSTANT TO TYPE-OUTPUT

     ELSE

          IF EXCEPTION-SWITCH NOT = 'ON'

               MOVE ERROR-CONSTANT TO TYPE-OUTPUT.
```

1. Notice the pairing of the IF's except the last unmatched IF.

2. If not an 'A' or a 'D' and EXCEPTION SWITCH is not 'ON,' then there is an error.

V. Nested IF's with NEXT SENTENCE

A. General Format

Note: NEXT SENTENCE means go to the beginning of the next sentence and skip processing of the rest of the current sentence.

```
IF CONDITION-1
     NEXT SENTENCE
ELSE
     IF CONDITION -2
          STATEMENT-2.
```

B. General notes on the format.

1. In the format, if condition-1 is true, then the NEXT SENTENCE statement takes processing control out of the sentence to the beginning of the next sentence.

2. NEXT SENTENCE at the end of a sentence is redundant.

C. Example of nested IF's with NEXT SENTENCE:

```
IF TYPE-INPUT NOT EQUAL 'SALE'
     NEXT SENTENCE
ELSE
     IF SALES-AMT IS GREATER THAN 100
          IF COMMISSION-CODE IS EQUAL 'YES'
               PERFORM COMPUTE-COMMISSION
          ELSE
               . . .
```

1. In the example, the test of being a sale is used with the NEXT SENTENCE command to avoid computations involved in computing commissions.

2. NEXT SENTENCE can be part of a more complex statement as in the following extension of the preceding example:

```
IF TYPE-INPUT NOT EQUAL 'SALE'
     MOVE 'YES' TO BILL-CODE
     NEXT SENTENCE
```

```
          IF SALES-AMT IS GREATER THAN 100
              IF COMMISSION-CODE . . .
```

VI. Multiple Level Nested IF's

A. General Format:

NOTE: As many as ten or even twenty levels of nested IF's are sometimes used. The example below illustrates three levels.

```
IF CONDITION-1
    STATEMENT-1
ELSE
    IF CONDITION-2
        STATEMENT-2
    ELSE
        IF CONDITION-3
            STATEMENT-3
        ELSE
            STATEMENT-4
```

B. Notes on general format:

1. The format is straightforward. If the condition-1 is true, then do statement-1, etc.
2. What is often not realized is that as soon as any one condition is true and its associated statements(s) is (are) processed, no more processing is done in that sentence; i.e., the conditions are tested as if they are mutually exclusive.
3. Only use this format for mutually exclusive conditions.

C. Example of multiple level nested IF's.

```
IF TYPE = 'A'
    PERFORM ADD-ROUTINE
ELSE
    IF TYPE = 'D'
        PERFORM DELETE-ROUTINE
```

```
ELSE

     IF TYPE = 'C'

          PERFORM CHANGE-ROUTINE

     ELSE

          PERFORM ERROR-ROUTINE.
```

1. The type in this example must be A, D, or C or it is an error. This is an example of a mutually exclusive condition.

2. Note that if a NEXT SENTENCE is inserted before each ELSE, there will be no effect on processing. There is an implicit NEXT SENTENCE in each IF statement in this format.

3. Do not use NEXT SENTENCE with this format, but <u>do</u> remember it is implied by the mutually exclusive requirement.

4. Study TM I as another example of nested IF's. Note the order of pairing. The last IF is paired with the first ELSE, the second last IF is paired with the second ELSE, etc. The order is further indicated by the indentation in the TM.

VII. Designing nested IF statements

A. Rule I: A nested IF is required when: A given condition must be tested <u>ONLY</u> when a previous condition has been tested <u>AND</u> alternative actions are required for one or more of the conditions.

B. Example of Rule I:

```
IF HOURLY-EMPLOYEE

     IF HOURS GREATER THAN 40

          PERFORM C010-CALCULATE-OUT-PAY

     ELSE

          PERFORM C020-CALCULATE-REG-PAY

ELSE

     PERFORM C030-CALCULATE-SAL-PAY
```

1. Note condition-I is a level 88 - a name actually assigned to a value.

2. Note the pairing of IF's and ELSE's. Here the inverse order rule applies as there are multiple IF conditions without an intervening ELSE.

3. Here rule 1 applies because "hours greater than 40" is only to be tested if an hourly-employee and the alternative actions of calculating overtime or regular pay must be decided.

C. Rule 2: A nested IF is required when:

A given condition must be tested

<u>ONLY</u>

when a previous condition has been tested

<u>AND</u>

one or more statements are to be executed

<u>BEFORE</u>

the second condition is tested.

D. Example of Rule 2:

```
IF COMMISSION-SALESMAN
      MOVE COMMISSION-CONSTANT TO COMMISSION-REPORT
      IF SALES-INPUT GREATER THAN COMMISSION-SALES-QUOTA
            MOVE OVER-QUOTA-CONSTANT TO
                  QUOTA-MESSAGE-REPORT.
```

(NOTE: In the example, if there is a commission base salesperson, then the commission is to be moved to the report. This satisfies the second part of this rule because this is done regardless of exceeding the sales quota.)

VIII. Steps in writing a complex nested IF statement from programming specifications

A. Study the programming specifications. (TM 2)

B. Determine which condition is most "inclusive;" i.e., applies to most transactions. (TM 3)

C. Steps in writing (TM 4)

1. Step 1: Write the most inclusive IF's conditions.

2. Example of Step 1:

```
IF DISCOUNT-CUSTOMER
```

3. Step 2: Write actions (may be further conditions) for most

inclusive IF's true condition.

4. Example of Step 2:

```
IF DISCOUNT-CUSTOMER
    IF SALES-INPUT GREATER MAXIMUM-LOW-DISC-SALES
```

5. Step 3: Write statements for second condition.

6. Example of Step 3:

```
IF DISCOUNT-CUSTOMER
    IF SALES-INPUT GREATER MAXIMUM-LOW-DISC-SALES
        COMPUTE DISCOUNT-AMT = SALES-INPUT * HIGH-
        PCT
```

7. Step 4: Write statements for CONDITION-2 not true.

8. Example of Step 4:

```
IF DISCOUNT-CUSTOMER
    IF SALES-INPUT GREATER MAXIMUM-LOW-DISC-SALES
        COMPUTE DISCOUNT-AMT = SALES-INPUT * HIGH-
        PCT
    ELSE
        COMPUTE DISCOUNT-AMT = SALES-INPUT * LOW-
        PCT
```

9. Step 5: Write statements when CONDITION-1 not true.

10. Example of Step 5:

```
IF DISCOUNT-CUSTOMER
    IF SALES-INPUT GREATER MAXIMUM-LOW-DISC-SALES
        COMPUTE DISCOUNT-AMT = SALES-INPUT * HIGH-
        PCT
    ELSE
        COMPUTE DISCOUNT-AMT = SALES-INPUT * LOW-
        PCT
ELSE
    MOVE NO-DISCOUNT-CONSTANT TO
    MESSAGE-REPORT.
```

IX. Special nested IF conditions

A. Some special conditions are difficult or impossible to program with nested IF's.

B. These special conditions and most nested IF's can be programmed or made simpler by grouping the complicated code into a submodule.

C. Example of submodule to simplify IF (TM 5)

X. Combined IF statements

A. General Format

```
IF condition-1
    AND/OR
    condition-2
        statement-1.
```

1. The words 'AND' and 'OR' are logical connectors to call for simultaneous test of two or more conditions.

2. With 'AND' - both condition-1 and condition-2 must be true before statement-1 is executed.

3. With 'OR' - if either condition-1 or condition-2 is true or if both are true, statement-1 is executed.

4. The "AND" is different from IF's in that all conditions are tested before any processing can take place

5. The "OR" is different from nested IF's in that each condition ranks equal to the other and the first true condition continues processing.

B. General example of combined IF's

Logical AND:

```
IF HOURS-WORKED
    GREATER 40
    AND
    HOURLY-EMPLOYEE
```

```
          PERFORM C00-OVERTIME-PAY-CALC
```

Logical OR:

```
IF HOURS-WORKED
     GREATER THAN 168
     OR
     HOURS-WORKED IS NEGATIVE
          PERFORM D100-ERR-ROUTINE
```

1. This is an example of a range test. It is not physically possible to work more than 168 hours in a week or negative hours, thus if either condition is true, the ERROR ROUTINE should be done.

XI. Complex IF statements with parenthesis

A. General format:

```
IF (condition-1
     OR
     condition-2)
     AND
     condition-3
     statement-1.
```

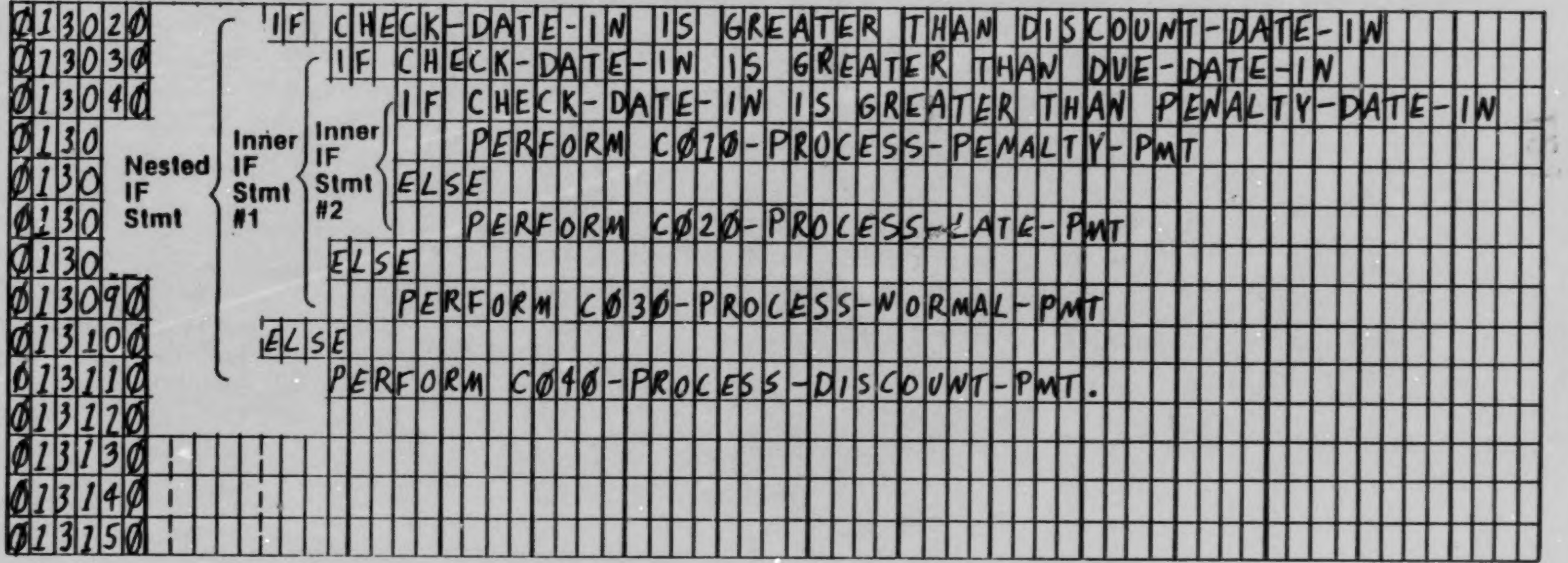
B. General Notes:

1. Although there is a defined sequence without using parenthesis, most often the parenthesis simplifies the coding.

2. The general rule is that the condition within the parenthesis is evaluated first.

C. Example of complex IF statements with parenthesis (TM 6)

Multiple Level Nested IF's with Inverted Pairing



```
013020 IF CHECK-DATE-IN IS GREATER THAN DISCOUNT-DATE-IN
013030   IF CHECK-DATE-IN IS GREATER THAN DUE-DATE-IN
013040     IF CHECK-DATE-IN IS GREATER THAN PENALTY-DATE-IN
0130           PERFORM C010-PROCESS-PENALTY-PMT
0130         ELSE
0130           PERFORM C020-PROCESS-LATE-PMT
0130       ELSE
013090       PERFORM C030-PROCESS-NORMAL-PMT
013100   ELSE
013110     PERFORM C040-PROCESS-DISCOUNT-PMT.
013120
013130
013140
013150
```

Example Programming Specifications for Writing A Nested IF

Programming Specifications:

1. If the sales to a customer are greater than $2,000 and the customer is entitled to a discount, the discount percentage is 18%.
2. If the sales to a customer are equal to or less than $2,000 and the customer is entitled to a discount, the discount percentage is 12%.
3. The customer is entitled to a discount if the value "D" is found in column 32 of the input record.
4. If the customer is not entitled to a discount, the message "NO DISCOUNT" should be printed on the report.

The Most Inclusive Condition

To determine which condition is most inclusive ask:

"Is there any reason to test this condition if the results of another condition are not known?"

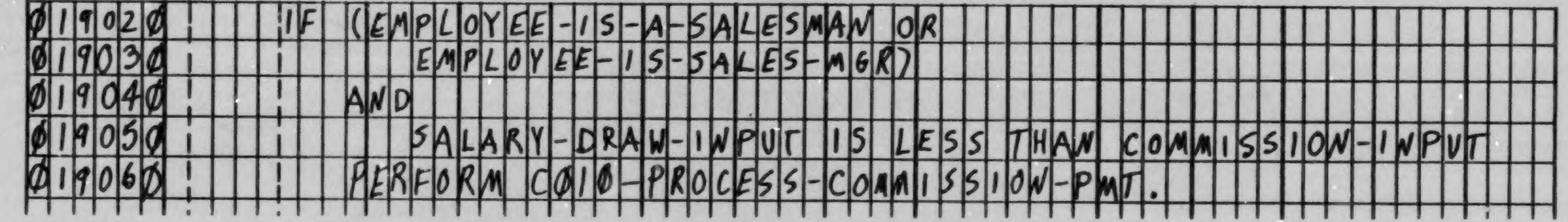
Summary of Steps in Writing Nested IF's

1. Write most inclusive IF (condition-1).
2. Write actions and any further conditions (condition-2) for most inclusive IF's true condition.
3. Write actions for further conditions (condition-2) true condition.
4. Write actions for further conditions (condition-2) not true condition.
5. Write actions for most inclusive IF's (condition-1) not true condition.

Use of Submodule to Simplify Complex IF Statement

```
013020     IF COMMISSION-SALESMAN
013030        PERFORM B001-CHECK-SALES
013040        ADD COMMISSION-WORK TO TOTAL-SALESMAN-SALARY-ACCUM
013050     ELSE
013060        ADD SALARY-INPUT TO TOTAL-SALESMAN-SALARY-ACCUM.
013070
013080
013090
013100 B001-CHECK-SALES.
013110
013120     IF SALES-INPUT IS GREATER THAN COMMISSION-QUOTA-CONSTANT
013130        COMPUTE COMMISSION-WORK = SALES-INPUT * .10
013140     ELSE
013150        COMPUTE COMMISSION-WORK = SALES-INPUT * .08.
```

Example of a Complex IF Simplified through Parenthesis



```
019020     IF (EMPLOYEE-IS-A-SALESMAN OR
019030           EMPLOYEE-IS-SALES-MGR)
019040        AND
019050           SALARY-DRAW-INPUT IS LESS THAN COMMISSION-INPUT
019060        PERFORM C010-PROCESS-COMMISSION-PMT.
```

Evaluation:

1. EMPLOYEE-IS-A-SALESMAN OR
   EMPLOYEE-IS-SALES-MGR
   AND
2. SALARY-DRAW-INPUT IS LESS THAN
   COMMISSION-INPUT

NESTED AND OTHER IF STATEMENTS

UNIT V

ASSIGNMENT SHEET 1: Simple Nested IF'S with Matched and Unmatched ELSE'S

1. Write a matched IF/ELSE sentence for the following edit specifications:

   Programming specification:

   > Check the transaction code for ADD, DEL, or CHG. If not one of the codes, perform an error routine. Otherwise, perform the respective addition, delete or change routine.

2. Write an unmatched IF/ELSE sentence for the following:

   Programming specifications:

   > Only for customer whose discount code is '2.' If the sales-code is 'A,' compute a discount of sales-amount multipled by discount-rate; otherwise, for nonsales-code 'A,' set discount equal to zero.

NESTED AND OTHER IF STATEMENTS

UNIT V

ASSIGNMENT SHEET 2: Semi-matched and NEXT SENTENCE IF Sentences

1. Write a semi-matched IF/ELSE sentence for the following pseudocode:

```
IF CODE-FIELD ≠ 1
    MOVE MESSAGE TO ERR-FIELD
ELSE
    IF TYPE = 2
        MOVE CONSTANT TO DISCOUNT-FIELD
    ELSE
        IF CLASS = 3
            PERFORM COMPUTE-DISCOUNT MODULE.
```

2. Write a NEXT SENTENCE type IF/ELSE sentence for the following pseudocode:

```
IF CLASS ≠ '79'
    NEXT SENTENCE
    IF SOPHOMORE
        DO COMPUTE GRADE-POINT-AVERAGE.
```

ASSIGNMENT SHEET 3: Multiple Level IF Sentences

1. Write a five-level multiple level IF statement for the following pseudocode: programming specification for a grade-edit program.

   Convert each of the following letter grades: A, B, C, D, F to 4, 3, 2, 1, and 0, respectively. If the field called grade is not A through F, set an error switch on.

ASSIGNMENT SHEET 4: Designing Nested IF's and Steps in Writing Nested IF's

1. Make up an application situation appropriate for use of Rule I for determining the necessity of a nested IF statement.

ASSIGNMENT SHEET 5: Special, Combined, and Complex IF Statements

1. Using the AND condition, write an example of a combined IF statement.

2. Using parenthesis, write a correct complex IF statement.

NESTED AND OTHER IF STATEMENTS

UNIT V

ANSWERS TO ASSIGNMENT SHEETS

Assignment Sheet 1

1.
```
IF T-CODE = 'ADD'
    PERFORM ADD-MODULE
ELSE
    IF T-CODE = 'DEL'
        PERFORM DELETE-MODULE
    ELSE
        IF T-CODE = 'CHG'
            PERFORM CHG-MODULE
        ELSE
            PERFORM ERR-MODULE.
```

2.
```
IF DISCOUNT-CODE = '2'
    IF SALE-CODE = 'A'
        COMPUTE DISCOUNT = SALES-AMOUNT
            *DISCOUNT-RATE
    ELSE
        MOVE ZERO TO DISCOUNT.
```

Assignment Sheet 2

1.
```
IF CODE NOT = 1
    MOVE MESSAGE TO ERR-FIELD
ELSE
    IF TYPE = 2
        MOVE CONSTANT TO DISCOUNT-FIELD
        IF CLASS = 3
```

```
                PERFORM COMPUTE-DISCOUNT-MODULE
2.  IF CLASS NOT = '79'
        NEXT SENTENCE
            IF SOPHOMORE
                PERFORM COMPUTE-GRADE-POINT-AVERAGE.
```

Assignment Sheet 3

```
1.  IF GRADE = 'A'
        MOVE '4' TO GRADE
    ELSE
        IF GRADE = 'B'
            MOVE '3' TO GRADE
        ELSE
            IF GRADE = 'C'
                MOVE '2' TO GRADE
            ELSE
                IF GRADE = 'D'
                    MOVE '1' TO GRADE
                ELSE
                    IF GRADE = 'F'
                        MOVE '0' TO GRADE
                    ELSE
                        MOVE 'X' TO ERR-SW.
```

Assignment Sheet 4

1. Example: A discount rate is only to be applied to class 1 customers who purchase over $1,000 and a different discount rate for class 1 customers is to be applied under $1,000.

Assignment Sheet 5

```
1.  IF SOPHOMORE
```

```
        AND
        G-P-A GREATER THAN 2.0
        PERFORM ADMIT-TO-B-SCHOOL-RTN.
2.  IF (SOPHOMORE
        AND
        G-P-A GREATER THAN 2.6)
        OR
        (TRANSFER STUDENT
        AND
        G-P-A GREATER THAN 3.0)
        PERFORM ADMIT-TO-B-SCHOOL-RTN.
```

NESTED AND OTHER IF STATEMENTS

UNIT V

EXAMINATION

1. Match the functional definitions in Column A with the corresponding terms in Column B.

Column A	Column B
____ a. Taken when IF is not true	1. Most inclusive statement
____ b. Proceed after this sentence	2. Matching IF/ELSE
____ c. Equal number of IF's and ELSE's	3. ELSE
____ d. IF contained within other IF statements	4. Nested IF
____ e. Applies to the most possible actions	5. NEXT SENTENCE

2. Write a matched nested IF statement for the following pseudocode specification.

```
If code is equal to 'B'
    do B-routine
else
    do err-routine
```

(NOTE: do is another way of saying perform.)

3. Write an unmatched IF statement for the following pseudocode specification.

```
If customer-type is one
    If trans-code equal 'P'
        compute balance = balance plus sales-amt
    else
        compute balance = balance minus payment
```

4. Write semi-matched nested IF/ELSE statements for the following pseudocode specifications.

```
If customer-type equals one
    move customer-description-one to
        output-type
else
    If customer-type equals two
        move customer-description-two
            to output-type
    else
        If exception-switch not on, move error-description to output-type.
```

5. Write a nested IF with use of NEXT SENTENCE for the following pseudocode specification.

```
If amount greater than credit limit
    next sentence
else
    If amount not less than discount-limit
        If discount-code equals "OK"
            compute discount = amount times discount-rate.
```

6. Write a multiple level nested IF for the following pseudocode specification.

```
If location-code equal one
    do Milwaukee-rtn
else
    if location-code equal two
        do Chicago-rtn
    else
        if location-code equal three
            do St-Louis-rtn
        else
```

```
            do error-rtn
```

7. State the two rules for designing nested IF statements.

8. List steps in writing a nested IF statement.

9. Explain how complex nested IF statements may be simplified.

10. Write a combined IF statement using parenthesis and correct logical operators for the following pseudocode specifications.

```
If (sex equal female
      and
      age greater than twenty-one)
      or
      (sex equal male
      and
      age greater than thirty-five)
            do insurance-discount-rtn.
```

NESTED AND OTHER IF STATEMENTS

UNIT V

ANSWERS TO EXAMINATION

1. a. 3
   b. 5
   c. 2
   d. 4
   e. 1

```
2.   IF CODE EQUAL 'B'

          PERFORM B-RTN

     ELSE

          PERFORM ERR-RTN.
```

```
3.   IF CUSTOMER-TYPE = '1'

          IF TRANS-CODE = 'P'

               COMPUTE BALANCE = BALANCE + SALES-AMT

          ELSE

               COMPUTE BALANCE = BALANCE - PAYMENT.
```

```
4.   IF CUSTOMER-TYPE = '1'

          MOVE CUSTOMER-DESCRIPTION-ONE TO

               OUTPUT-TYPE

     ELSE

          IF CUSTOMER-TYPE = '2'

               MOVE CUSTOMER-DESCRIPTION-TWO TO

                    OUTPUT-TYPE

          ELSE

               IF EXCEPTION-SWITCH NOT = 'ON'

                    MOVE ERROR-DESCRIPTION

                         TO OUTPUT-TYPE.
```

```
5.    IF AMOUNT GREATER THAN CREDIT-LIMIT
              NEXT SENTENCE
      ELSE
              IF AMOUNT NOT LESS THAN
                      DISCOUNT-LIMIT
                      IF DISCOUNT-CODE = 'OK'
                              COMPUTE DISCOUNT = AMOUNT X DISCOUNT-RATE.
```

```
6.    IF LOCATION-CODE = '1'
              PERFORM MILWAUKEE-RTN
      ELSE
              IF LOCATION-CODE = '2'
                      PERFORM CHICAGO-RTN
              ELSE
                      IF LOCATION-CODE = '3'
                              PERFORM ST-LOUIS-RTN
                      ELSE
                              PERFORM ERROR-RTN.
```

7. Rule 1: A nested IF is required when a given condition must be tested only when a previous condition has been tested and alternative actions are required.

   Rule 2: A nested IF is required when a given condition must be listed only when a previous condition has been tested and statement(s) are to be executed before the second condition is tested.

8. 1. Write most inclusive IF
   2. Write action (or condition) for true part of 1
   3. Write action for second condition's true part
   4. Write action for second condition's not true part
   5. Write actions for not true part of item 1.

   (NOTE: This solution is based upon Shelly and Cashman's method and is specific to their example.)

9. Can simplify complex nested IF by writing part of conditionals with a submodule and performing the submodule from within the IF.

```
10.   IF (SEX = 'F'
           AND
           AGE GREATER THAN 21)
           OR
           (SEX = MALE
           AND
           AGE GREATER THAN 35)
                PERFORM INSURANCE-DISCOUNT-RTN.
```

CONTROL BREAKS

UNIT VI

UNIT OBJECTIVE

After studying this unit, the student should be able to utilize control break processing techniques.

SPECIFIC OBJECTIVES

After studying this unit, the student should be able to:

1. Match terms and definitions.
2. Recognize correct statements of background information on control breaks.
3. Analyze processes in initial record processing in a control break program.
4. Determine which modules share in processing of an initial detail record in a control break program.
5. Walk through the steps in processing of a second detail record in control break programs.
6. Explain setting of accumulators and other steps in processing of a control break in control break programs.
7. Explain logic of the steps in final control break processing in a control break program.
8. Recognize correct statements in overview of I/O in a sample control break program.
9. Explain programming specifications for a sample control break program.
10. Recognize correct statements based on IPO charts for a sample program.
11. Recognize correct statements based on pseudocode specifications for a sample program.
12. Explain details for the coding of a sample control break program.

CONTROL BREAKS

UNIT VI

SUGGESTED ACTIVITIES

I. Instructor

A. Make transparencies; prepare lectures, using this curriculum guide and supplemental materials.

B. Provide students with objectives; discuss objectives.

C. Make parallel assignments for reading and programming from the text(s).

D. Provide students with information and assignment sheets; discuss these sheets.

E. Provide students with additional supplementary information and exercises.

F. Give examination.

II. Student

A. Read objectives.

B. Study information sheets and text assignments.

C. Complete assignment sheets and supplementary exercises.

D. Complete examination.

INSTRUCTIONAL MATERIALS

I. Unit content

A. Objectives

B. Information sheets

C. Handouts

1. HO 1: The Program Plan - Programming Specifications

2. HO 2: Pseudocode for Control Inventory Report Module

3. HO 3: Pseudocode Specifications for Detail Record Processing Module

4. HO 4: Pseudocode Specifications for Processing Department Changes Module

5. HO 5: Coding for Control Inventory Report Module

D. Transparency masters

1. TM 1: Example of Control Break Input
2. TM 2: Report Subtotal Examples of Printing During Control Breaks
3. TM 3: Example Card Format for a Control Break Program for a Simple Inventory
4. TM 4: Initial Processing Steps in Control Break Processing
5. TM 5: Processing of First Detail Record in Control Break Programs
6. TM 6: Processing of a Detail Record in Control Break Programs
7. TM 7: Processing an Actual Control Break in Control Break Programs
8. TM 8: Final Total Processing
9. TM 9: Input Record Format for Sample Control Break Program
10. TM 10: Program Output for Sample Control Break Program
11. TM 11: IPO Chart and Hierarchy Chart for the Sample Program (Part 1 and 2)
12. TM 12: IPO Chart for Process Detail Records Module
13. TM 13: Hierarchy Chart for Process Detail Records Module
14. TM 14: IPO Chart for Process Department Changes Module
15. TM 15: IPO Chart for Print the Final Totals Module
16. TM 16: IPO Chart for Print the Headings Module
17. TM 17: Pseudocode Specifications for Final Total Printing
18. TM 18: Pseudocode for Printing the Headings

E. Assignment sheets

1. AS 1: Terms and Background Information on Control Breaks
2. AS 2: Initial Detail and Second Detail Processing in Control Break Programs

3. AS 3: Actual Control Break Processing and Final Total Control Break Processing

4. AS 4: A Sample Control Break Program

F. Answers to assignment sheets

G. Job Sheet 1: Writing a Payroll Validation Report Using Control Break Logic

1. Figure JS 1-1: Input Card Layout for Payroll Validation Report Program

2. Figure JS 1-2: Sample Control Hierarchy Chart for Payroll Validation Report Program

3. Figure JS 1-3: Prewritten Divisions for Payroll Validation Report

H. Job Sheet 1: Appendix: Sample Application, an Inventory List by Branch Location Report Application

1. Figure JS 1-4: Input Layout for Inventory by Branch Location

2. Figure JS 1-5: Hierarchy Chart for Inventory Report by Branch Location Program

3. Figure JS 1-6: Program Listing of Sample Application for Inventory by Branch (Parts 1 and 2)

4. Figure JS 1-7 Sample Output of Inventory Listing by Branch Location (Parts 1 and 2)

I. Answers to Job Sheet 1

1. Complete Solution to Payroll Validation Report (Parts 1, 2, 3, and 4)

J. Examination

K. Answers to examination

II. References

A. Shelly, Gary B. and Thomas J. Cashman. "Control Breaks" in Introduction to Structured COBOL Programming. Brea, CA: Anaheim Publishing Company, 1977.

CONTROL BREAKS

UNIT VI

INFORMATION SHEET

I. Terms and functional definitions

   A. **Key field** - field containing the record ID, used to order sequential files

   B. **Control break** - a change in the key field(s) of the input file, generally causes printing of a summary total line

   C. **Department accumulator** - field used to accumulate totals for a department key field

   D. **Final total accumulator** - field used to accumulate totals for the entire report

   E. **COMPUTATIONAL-3** - packed decimal internal format, used for computation of signed decimal fields, abbreviated COMP-3

   F. **Trailer card** - special card at end of deck used to control the end of job processing

II. Background information on control breaks

   A. Explanation and example

      1. Explanation - a control break occurs when there is change in the key field of the input file

      2. A key field is normally the record ID, and the file is sorted on the field.

      3. A control break occurs at ID's 2010, 3011, and 4012. (TM 1)

   B. Reports based on control breaks

      1. When a control break occurs, normally a subtotal or total is printed on a report.

      2. Example of report subtotal (TM 2)

   C. Inputs to control breaks

      1. Example of an input (TM 3)

      2. Note in the example that the record key in Department Number and file is in sorted order based on the key.

3. A control break is computed whenever the Department Number changes.

III. Initial record processing in control break programs (TM 4)

A. Steps

1. Read first record into input area.

2. Move Department Number from first record to a field called COMPARE-AREA.

(NOTE: COMPARE-AREA field is used to determine if a control break has occurred.)

IV. Processing of a detail record in control break programs (TM 5)

A. Steps are keyed to TM 5.

1. The first step is to compare Department Number of record in input area to the Department Number in the COMPARE-AREA; if they are equal, it is not a control break. This is called a detail record rather than a control break record.

2. Print headings for first page.

3. Move data from input record to the output print area.

4. Calculate Inventory Value.

5. Move Inventory Value to print area.

6. Accumulate running totals of inventory by adding the Inventory Value just calculated to department total of inventory, Department Total Accumulator, and to the overall report total of inventory, Final Total Accumulator.

7. Print the detail line .

V. Processing of a detail record in control break programs - second record (TM 6)

A. Steps are keyed to TM 6.

1. Read second record.

2. Compare Department Number in COMPARE-AREA to Department Number in the input record. As they are equal, it is not a control break.

3. Move input fields to output area.

4. Calculate Inventory Value.

5. Move Inventory Value to output area.

6. Accumulate Inventory Value to Department and Final Accumulators.

7. Print second record.

VI. Processing of an actual control break in control break programs (TM 7)

A. Steps are keyed to TM 7.

1. Read third record.

2. As Department Number in the third record is not equal to Department Number in the compare area, a control break has occurred.

3. Move Department Number from compare area to the Department Total line to set up proper number for the break.

4. Move Department Accumulator to the Department Total line.

5. Print the Department Total line.

6. Move zero to Department Accumulator.

7. Move Department Number from the input record to the compare area.

(NOTE: Next, process third Detail Record as was done for second record.)

VII. Final total control break processing in control break programs (TM 8)

A. Final total can occur in three ways

1. At end of file (card reader empty)

2. End of file sensed

3. Trailer card record is read

(NOTE: A trailer card is a special card at the end of the deck used for control purposes.)

B. Department Total line for last department is then printed.

C. Print Final Total line after moving Total Accumulator to the line.

VIII. Sample control break program - overview

A. Input (TM 9) - note same input of inventory fields as in preceding example.

B. Output of sample program (TM 10)

IX. Planning the sample control break program - programming specifications in HO 1 plan the program

X. Sample control break program - IPO Charts

A. IPO Chart - Control Inventory Report (TM 11)

B. IPO Chart - Process Detail Record (TM 12 and 13)

C. IPO Chart - Process Department Changes(TM 14)

D. IPO Chart - Print Final Totals (TM 15)

E. IPO Chart - Print the Headings (TM 16)

XI. Sample control break program - pseudocode specifications

A. Pseudocode specifications - Control Inventory Report Module (HO 2)

1. References in "REF." column are to other modules.

2. ENDIF and ENDPERFORM are currently for documentation purposes only. The new standard is expected to implement these commands.

3. PERFORM UNTIL (in this format) is also expected in the new standard as a PERFORM WHILE command. Also known as the "perform in place" command, it says to do whatever follows rather than go to another paragraph(s).

B. Pseudocode specifications - process detail record (HO 3)

1. Note that after Inventory Value is calculated in this module, it is also added to both the Department Total and Final Total Accumulators; alternatively, Final Total can be updated when there is a Department Break.

2. For line count incrementation, some programmers prefer to count (increment) by the number of lines printed plus the number of lines skipped.

C. Pseudocode specifications - process department change (HO 4)

1. The function of this module is to process the actual control break.

2. The control break information consists of the Department Totals.

3. Note particularly the setting of the Department Accumulator field to zero. This is necessary to start a new total for the next department.

D. Pseudocode specifications - print final totals (TM 17)

1. This is sometimes considered a second level control break; i.e., conceptually the company total is simply a total of the department totals; therefore, it is a second level total and break.

E. Pseudocode specifications - print headings (TM 18)

1. This is a semi-standard module which is used in many programs.

2. The logic in this module has been discussed in previous programs.

XII. Sample control break program coding

A. Data Division Coding

1. If the record key is always numeric, use COMP-3 format.

2. If the record key may have spaces, letters, dashs, etc., use an alphanumeric picture clause (PIC XXX format).

B. Procedure Division sample coding for Control Inventory Report Module (HO 5)

HO 1

The Program Plan-Programming Specifications

Programming Specifications

SUBJECT Inventory Report	**DATE** Oct 10	**PAGE 1 OF 1**
TO Programmer	**FROM** Systems Analyst	

A program is to be written to prepare an Inventory Report. The formats for the input card file and the printer spacing chart are included as a part of this narrative. The program should include the following processing.

1. The program should read the input cards and create the Inventory Report as per the format illustrated on the printer spacing chart. The report shall contain the Department Number, the Part Number, the Item Description, the Quantity On Hand, the Unit Cost, and the Inventory Value.
2. Calculate the Inventory Value by multiplying the Quantity On Hand by the Unit Cost.
3. Print headings on the first page of the report. The heading is to contain the current date and the page number.
4. Include report and column headings on subsequent pages. Fifty-five detail and total lines are to appear on each page before skipping to a new page.
5. Print one line on the report for each card which is read. The lines are to be single spaced.
6. Edit report as illustrated on the printer spacing chart.
7. Department totals are to be printed when there is a change in Department Number. A final total of the Inventory Value is to be printed after all cards have been processed. Spacing for the total lines should correspond to the printer spacing chart.
8. Write program in COBOL.

HO 2

Pseudocode for Control Inventory Report Module

Pseudocode Specifications

DATE: June 19	PROGRAMMER: John Smith	PROGRAM: Inventory Report
MODULE FUNCTION: Control the Inventory Report	REF: A000	MODULE NAME: Control Inventory Report

PSEUDOCODE / REF:

```
Open the files
Read an input record
IF there is a record
    Move department number to compare area
PERFORM UNTIL there are no more records
        IF department number not =
                department number in
                compare area
            Process department change          B010
        ENDIF
        Process detail records                 B000
        Read an input record
    ENDPERFORM
    Process department change                  B010
    Print final totals                         B020
ENDIF
Close files
Stop run
```

FILES, RECORDS, FIELDS REQUIRED

- Inventory Input File
- Inventory Report File
- Input area
    - Department number
    - Part number
    - Description
    - Quantity
    - Unit cost
- More records indicator
- Department number compare area
- No more records indicator

HO 3

Pseudocode
Specifications for Detail Record Processing Module

Pseudocode Specifications

DATE: June 19	**PROGRAMMER:** John Smith	**PROGRAM:** Inventory Report
MODULE FUNCTION: Process the Detail Records	**REF:** B000	**MODULE NAME:** Process Detail Record

PSEUDOCODE | **REF:**

```
If number of lines printed is = or >page size or
        this is the first page
    Print the headings                              C000
ENDIF
Clear the printer area
Move department number, part number,
        description, quantity, unit cost to
        printer output area
Calculate inventory value = quantity x unit cost
Move inventory value to department total,
        add final total
Write a line on the report
Set spacing for single spacing
Add 1 to the number of lines printed
```

FILES, RECORDS, FIELDS REQUIRED

- Number of lines printed counter
- Page size constant
- First page indicator
- Printer output area
    - Department number
    - Part number
    - Description
    - Quantity
    - Unit cost
    - Inventory value
- Input area
    - Department number
    - Part number
    - Description
    - Quantity
    - Unit cost
- Inventory value work area
- Department total accumulator
- Final total accumulator
- Space control field
- Single space control character

HO 4

Pseudocode Specifications for Processing Department Changes Module

Pseudocode Specifications

DATE: June 19	**PROGRAMMER:** John Smith	**PROGRAM:** Inventory Report
MODULE FUNCTION: Process Department Changes	**REF:**	**MODULE NAME:** Process Dept Change

PSEUDOCODE

```
Move previous department number to department
        total line
Move department total to department total line
Write the department total line on the report
Set spacing for triple spacing
Add 4 to the number of lines printed
Set the department total accumulator to zero
Move the new department number to the
        compare area
```

REF:

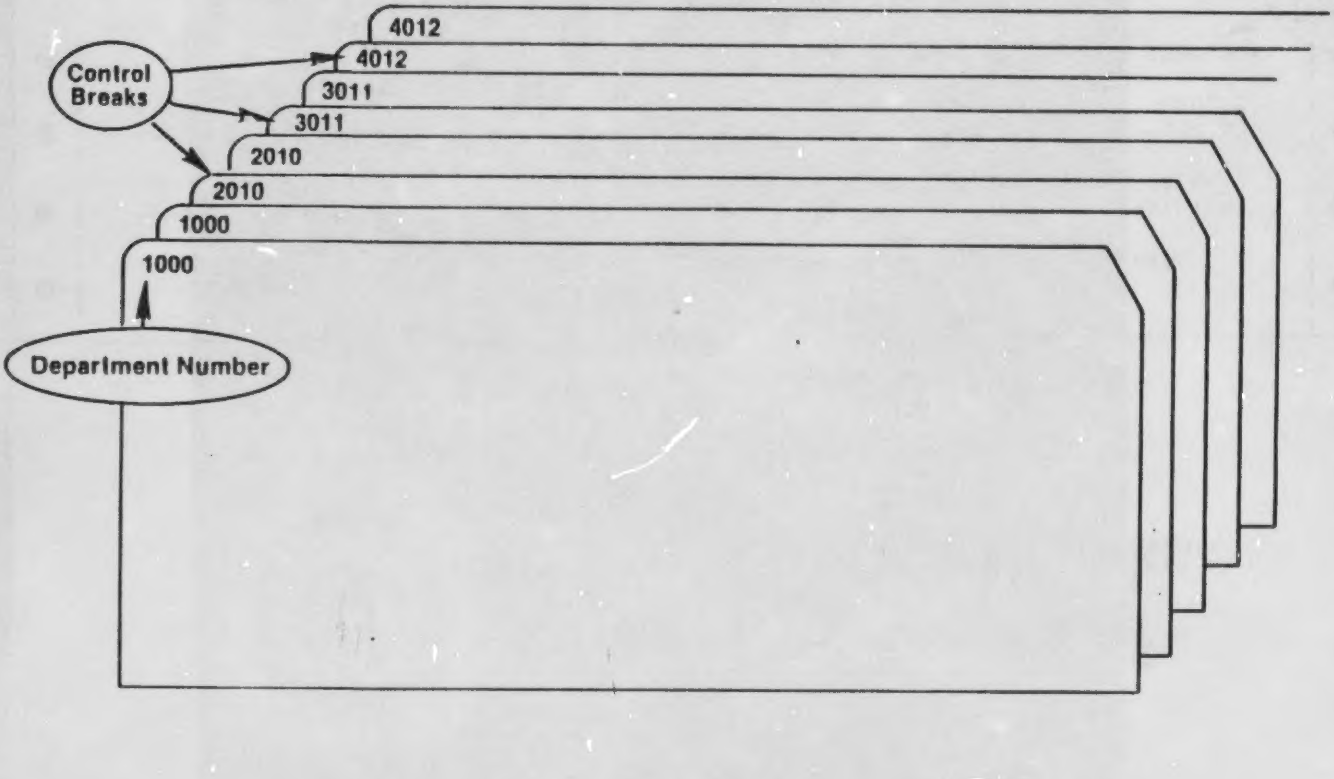
FILES, RECORDS, FIELDS REQUIRED

- Previous department number from compare area
- Department total accumulator
- Department total line
- Number of lines printed counter
- Triple spacing control character
- Space control field
- New department number from input record
- Department number compare area

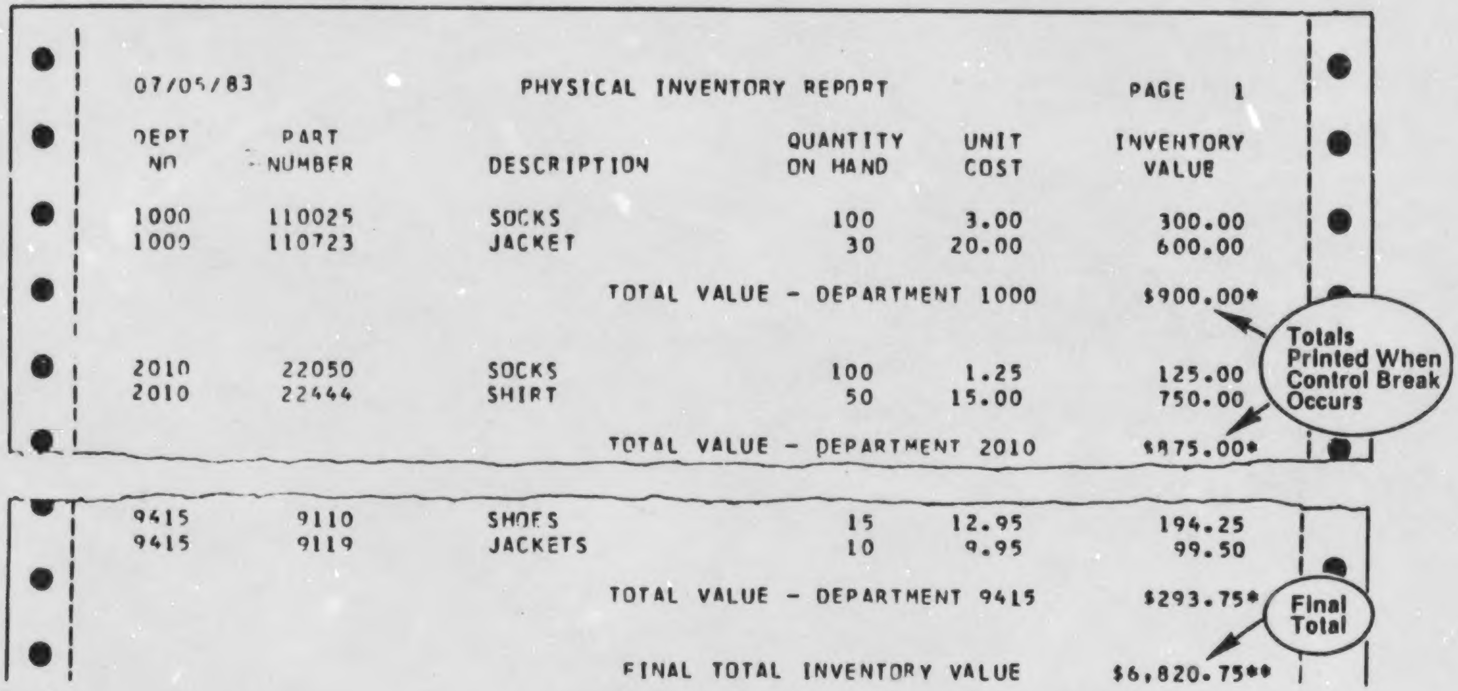
Coding for Control Inventory Report Module

```
009190 A000-CONTROL-INVENTORY-REPORT.
009200
010010     OPEN INPUT  INVENTORY-INPUT-FILE
010020          OUTPUT INVENTORY-REPORT-FILE.
010030     READ INVENTORY-INPUT-FILE
010040         AT END
010050             MOVE 'NO ' TO ARE-THERE-MORE-RECORDS.
010060     IF THERE-IS-A-RECORD
010070       MOVE DEPT-NUMBER-INPUT TO PREVIOUS-DEPT-NUMBER
010080       PERFORM A001-COMPARE-AND-PROCESS
010090             UNTIL THERE-ARE-NO-MORE-RECORDS
010100       PERFORM B010-PROCESS-DEPT-CHANGE
010110       PERFORM B020-PRINT-FINAL-TOTALS.
010120     CLOSE INVENTORY-INPUT-FILE
010130           INVENTORY-REPORT-FILE.
010140     STOP RUN.
010150
010160
010170
010180 A001-COMPARE-AND-PROCESS.
010190
010200     IF DEPT-NUMBER-INPUT IS NOT = PREVIOUS-DEPT-NUMBER
011010       PERFORM B010-PROCESS-DEPT-CHANGE.
011020     PERFORM B000-PROCESS-DETAIL-RECORD.
011030     READ INVENTORY-INPUT-FILE
011040         AT END
011050             MOVE 'NO ' TO ARE-THERE-MORE-RECORDS.
```

Example of Control Break Input


4012
4012
Control Breaks
3011
3011
2010
2010
1000
1000
Department Number

Report Subtotal Examples of Printing During Control Breaks



```
07/05/83                    PHYSICAL INVENTORY REPORT                   PAGE   1

DEPT      PART                                     QUANTITY    UNIT      INVENTORY
 NO      NUMBER          DESCRIPTION               ON HAND     COST        VALUE

1000     110025          SOCKS                         100     3.00        300.00
1000     110723          JACKET                         30    20.00        600.00

                              TOTAL VALUE - DEPARTMENT 1000               $900.00*

2010      22050          SOCKS                         100     1.25        125.00
2010      22444          SHIRT                          50    15.00        750.00

                              TOTAL VALUE - DEPARTMENT 2010               $875.00*

9415       9110          SHOES                          15    12.95        194.25
9415       9119          JACKETS                        10     9.95         99.50

                              TOTAL VALUE - DEPARTMENT 9415               $293.75*

                               FINAL TOTAL INVENTORY VALUE             $6,820.75**
```

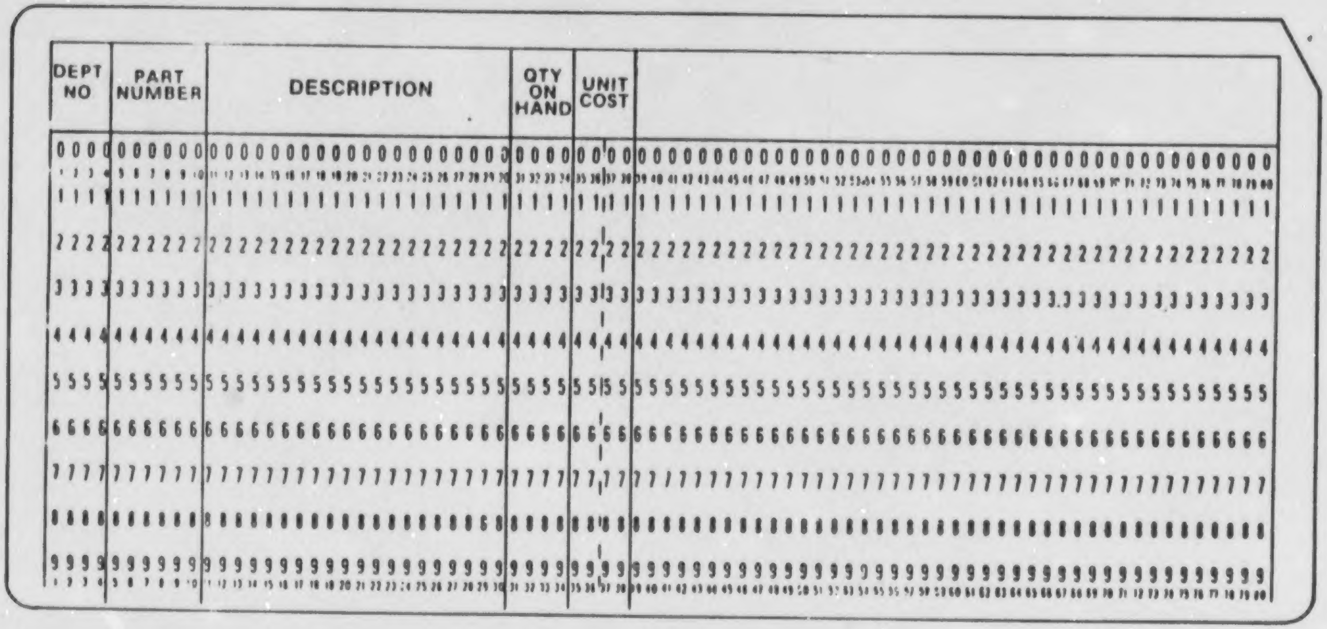
Totals Printed When Control Break Occurs

Final Total

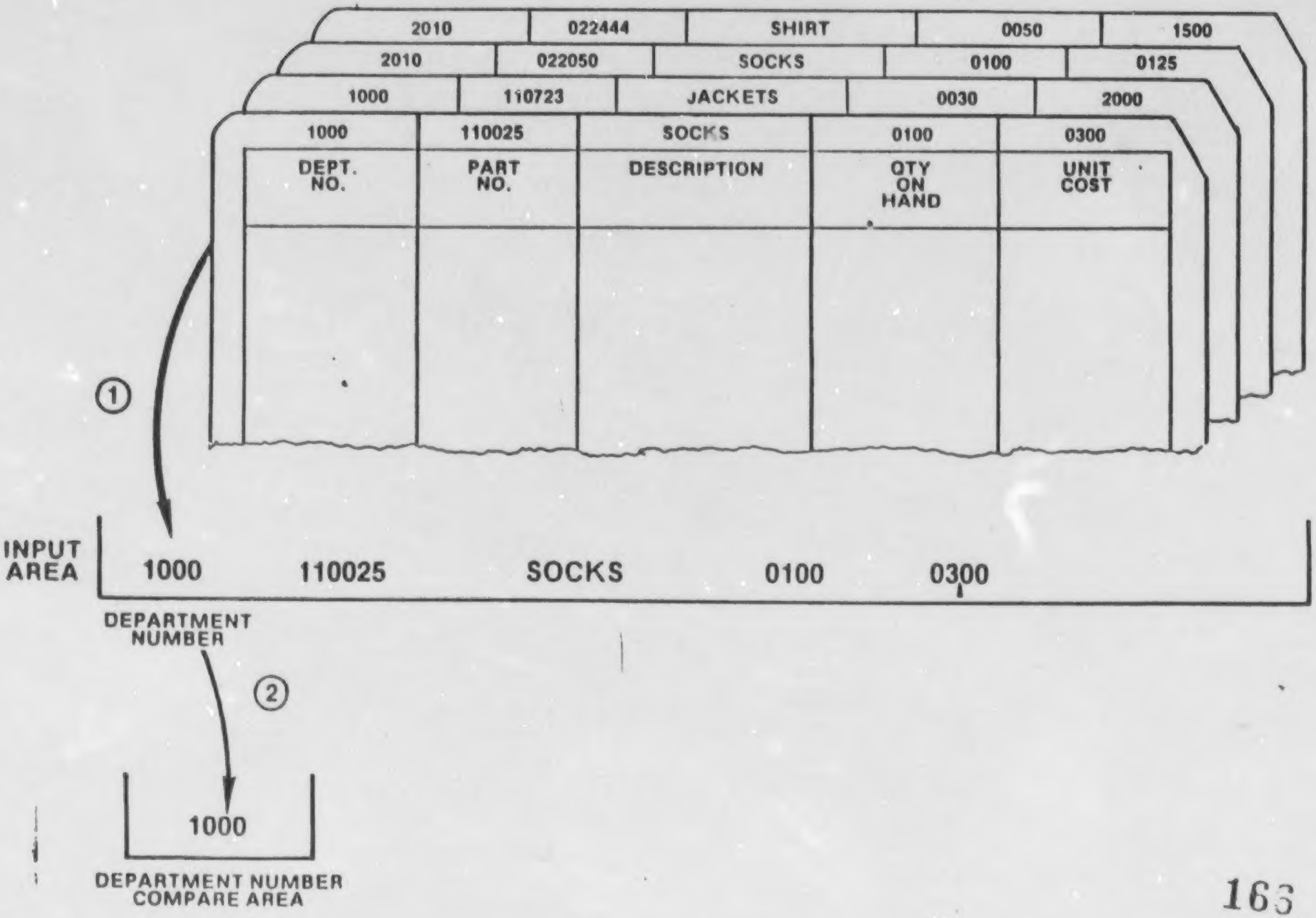
TM 2

TM 3

Example Card Format for a Control Break Program for a Simple Inventory


DEPT NO
PART NUMBER
DESCRIPTION
QTY ON HAND
UNIT COST

Initial Processing Steps in Control Break Processing



2010 022444 SHIRT 0050 1500

2010 022050 SOCKS 0100 0125

1000 110723 JACKETS 0030 2000

1000 110025 SOCKS 0100 0300

DEPT. NO.	PART NO.	DESCRIPTION	QTY ON HAND	UNIT COST

①

INPUT AREA 1000 110025 SOCKS 0100 0300

DEPARTMENT NUMBER

②

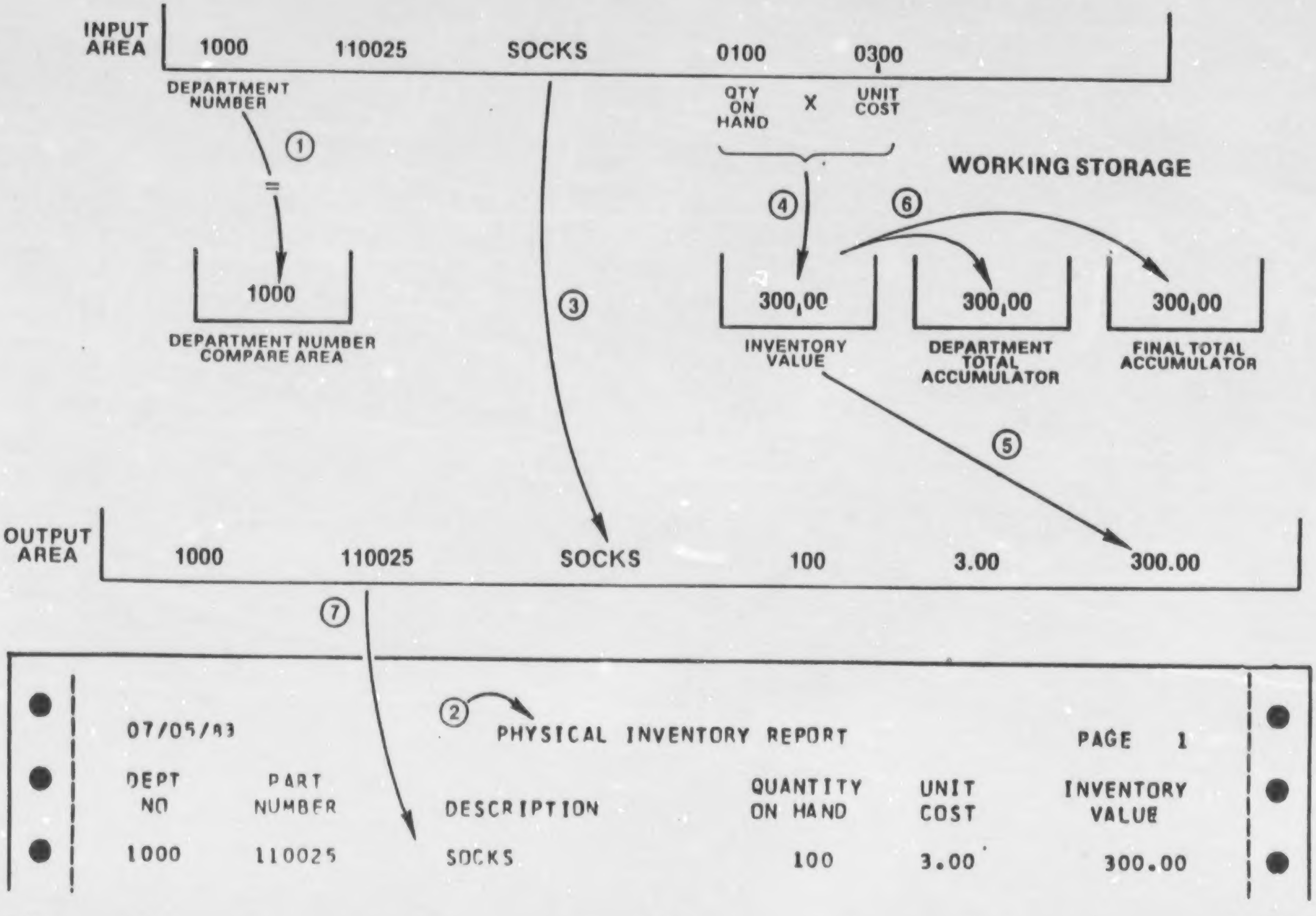
1000

DEPARTMENT NUMBER COMPARE AREA

TM 4

TM 5

Processing of First Detail Record in Control Break Programs



INPUT AREA 1000 110025 SOCKS 0100 0300

DEPARTMENT NUMBER ① =

1000

DEPARTMENT NUMBER COMPARE AREA

③

QTY ON HAND X UNIT COST

④ ⑥

WORKING STORAGE

300,00 INVENTORY VALUE

300,00 DEPARTMENT TOTAL ACCUMULATOR

300,00 FINAL TOTAL ACCUMULATOR

⑤

OUTPUT AREA 1000 110025 SOCKS 100 3.00 300.00

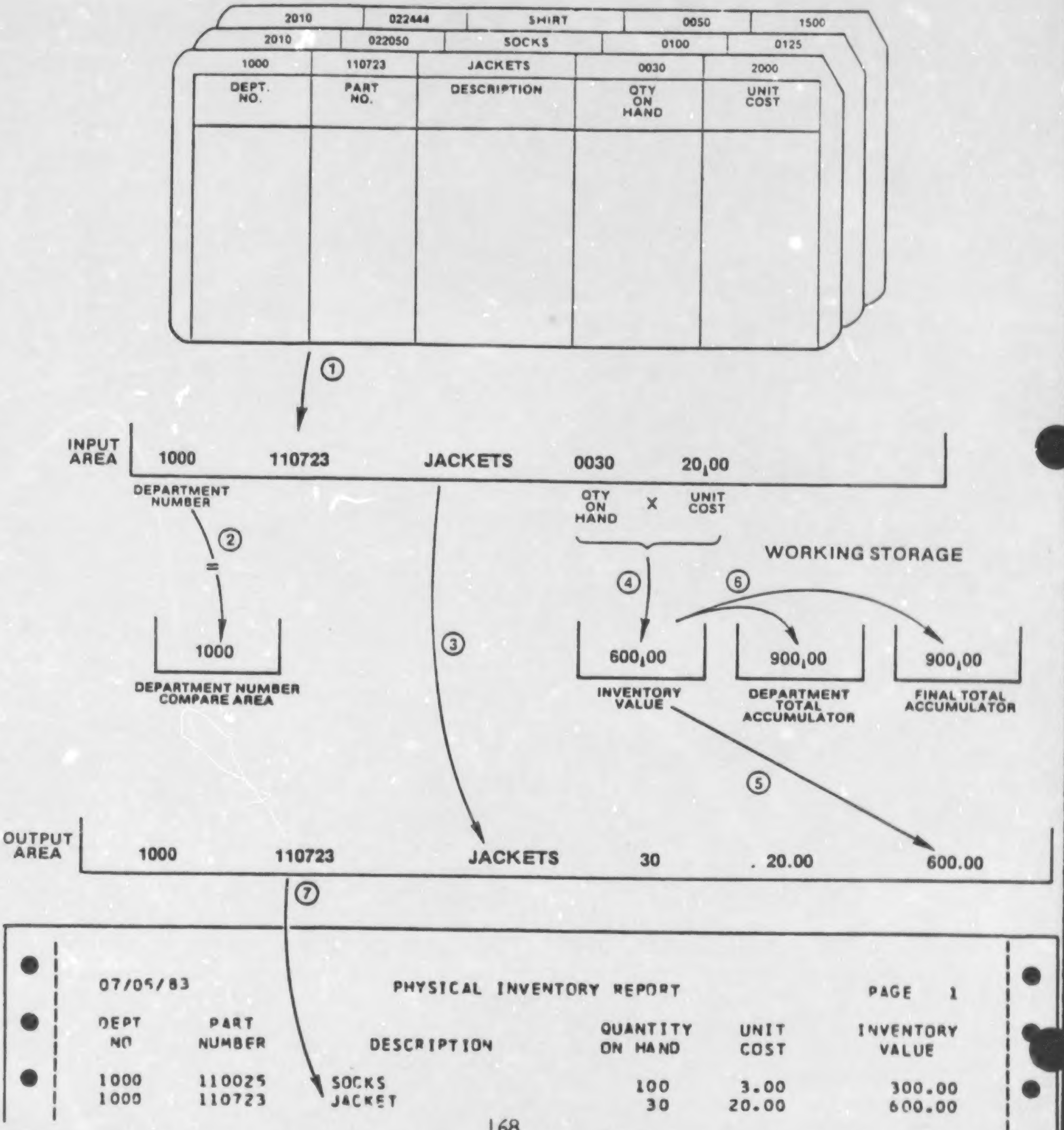
⑦

②

```
07/05/83                    PHYSICAL INVENTORY REPORT                        PAGE   1
DEPT       PART                                   QUANTITY      UNIT        INVENTORY
 NO       NUMBER        DESCRIPTION               ON HAND       COST          VALUE
1000      110025        SOCKS                        100        3.00         300.00
```

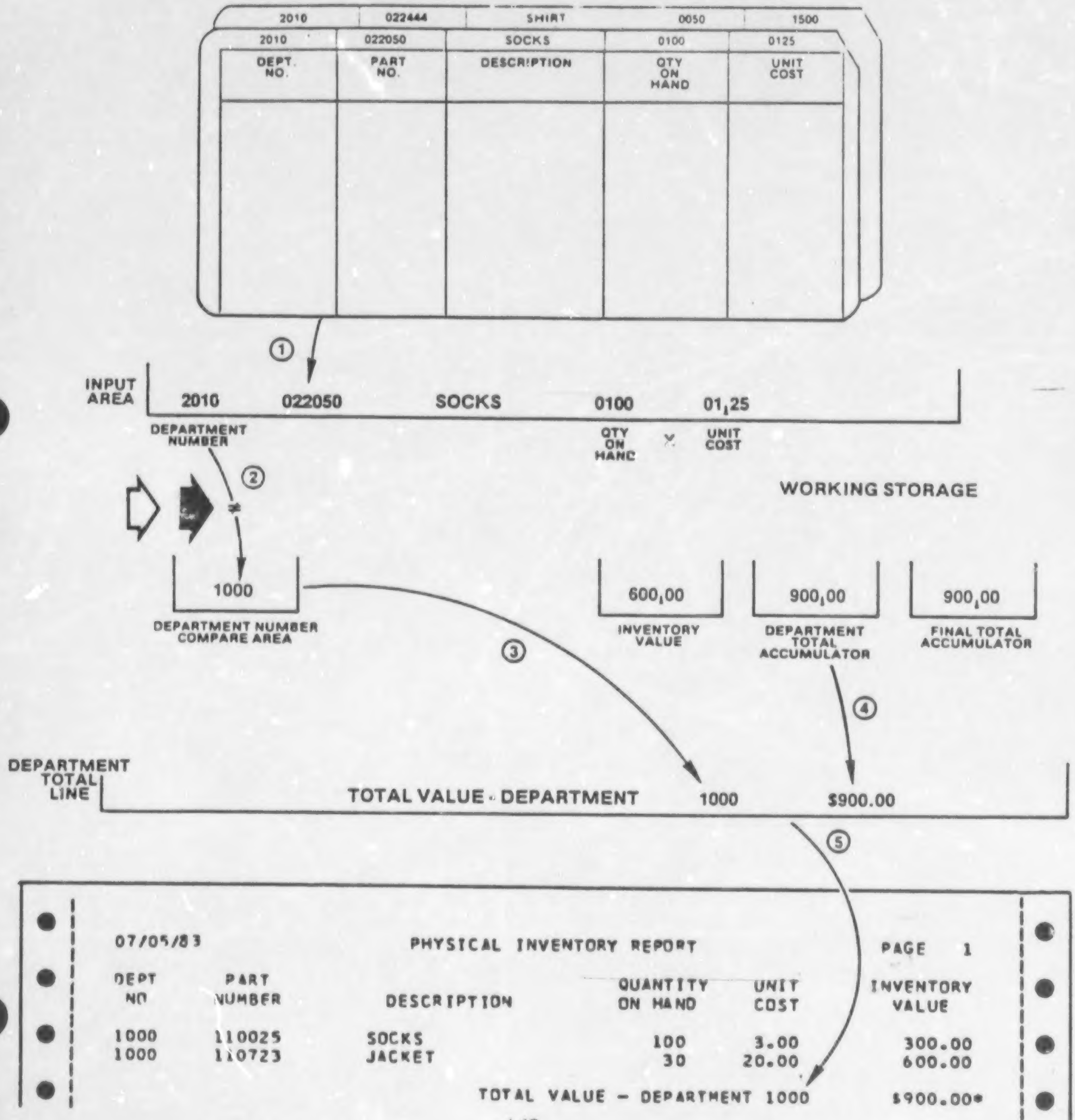
TM 6

Processing of a Detail Record in Control Break Programs


2010
022444
SHIRT
0050
1500
2010
022050
SOCKS
0100
0125
1000
110723
JACKETS
0030
2000
DEPT. NO.
PART NO.
DESCRIPTION
QTY ON HAND
UNIT COST
①
INPUT AREA
1000
110723
JACKETS
0030
20,00
DEPARTMENT NUMBER
②
=
1000
DEPARTMENT NUMBER COMPARE AREA
③
QTY ON HAND
X
UNIT COST
WORKING STORAGE
④
⑥
600,00
INVENTORY VALUE
900,00
DEPARTMENT TOTAL ACCUMULATOR
900,00
FINAL TOTAL ACCUMULATOR
⑤
OUTPUT AREA
1000
110723
JACKETS
30
. 20.00
600.00
⑦

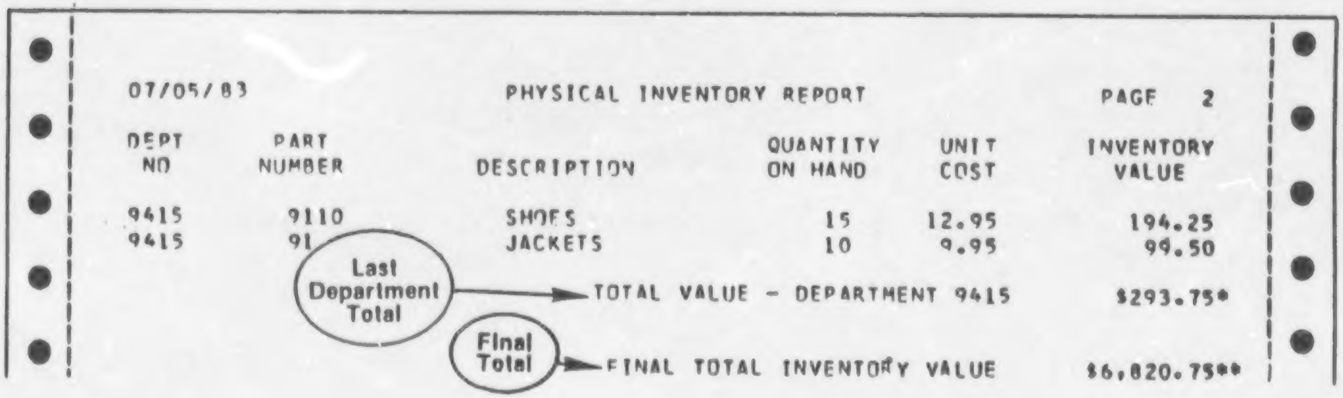
```
07/05/83                 PHYSICAL INVENTORY REPORT                    PAGE   1

DEPT       PART                                QUANTITY     UNIT        INVENTORY
 NO       NUMBER          DESCRIPTION          ON HAND      COST          VALUE

1000      110025     SOCKS                         100       3.00         300.00
1000      110723     JACKET                         30      20.00         600.00
```

Processing an Actual Control Break in Control Break Programs


2010
022444
SHIRT
0050
1500
2010
022050
SOCKS
0100
0125
DEPT. NO.
PART NO.
DESCRIPTION
QTY ON HAND
UNIT COST
①
INPUT AREA
2010
022050
SOCKS
0100
01,25
DEPARTMENT NUMBER
QTY ON HANC
×
UNIT COST
②
≠
WORKING STORAGE
1000
DEPARTMENT NUMBER COMPARE AREA
600,00
INVENTORY VALUE
900,00
DEPARTMENT TOTAL ACCUMULATOR
900,00
FINAL TOTAL ACCUMULATOR
③
④
DEPARTMENT TOTAL LINE
TOTAL VALUE - DEPARTMENT
1000
$900.00
⑤

```
07/05/83                     PHYSICAL INVENTORY REPORT                       PAGE    1

DEPT      PART                                       QUANTITY     UNIT       INVENTORY
 NO      NUMBER           DESCRIPTION                ON HAND      COST         VALUE

1000     110025        SOCKS                             100       3.00        300.00
1000     110723        JACKET                             30      20.00        600.00
                                  TOTAL VALUE - DEPARTMENT 1000               $900.00*
```

Final Total Processing

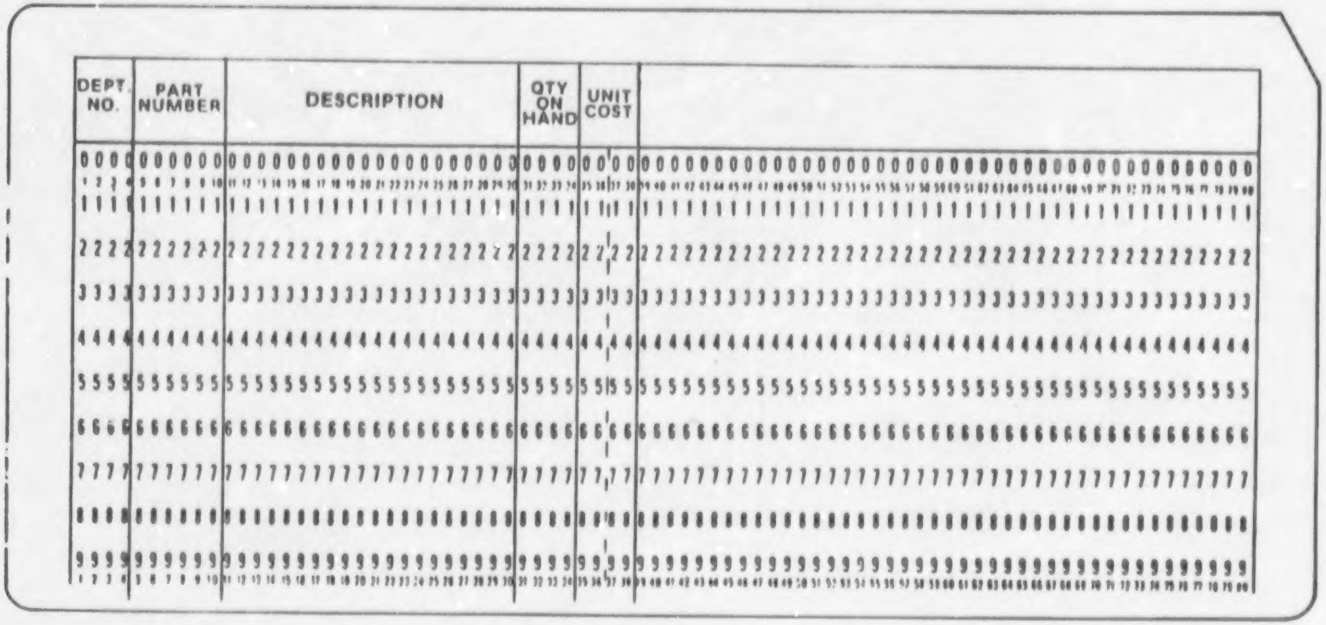


```
07/05/83                    PHYSICAL INVENTORY REPORT                     PAGE   2

DEPT      PART                                 QUANTITY     UNIT       INVENTORY
 NO       NUMBER         DESCRIPTION           ON HAND      COST         VALUE

9415       9110           SHOES                   15        12.95        194.25
9415       91             JACKETS                 10         9.95         99.50
                          TOTAL VALUE - DEPARTMENT 9415                 $293.75*

                            FINAL TOTAL INVENTORY VALUE               $6,820.75**
```

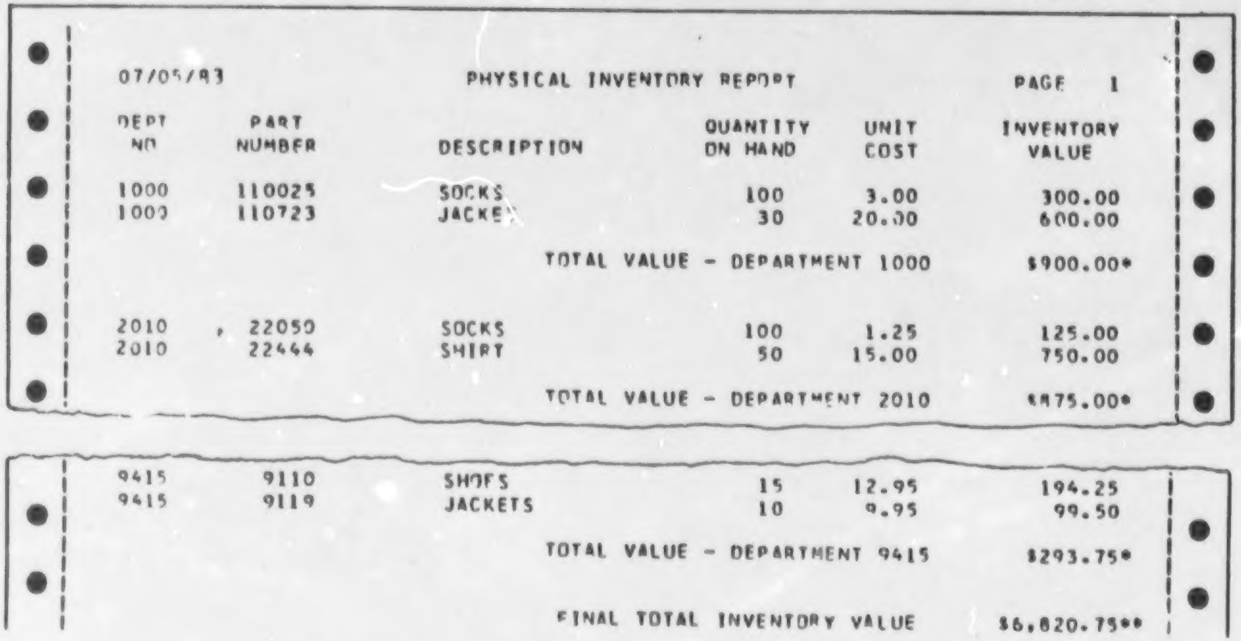
Last Department Total
Final Total

TM 9

Input Record Format for Sample Control Break Program


DEPT. NO.
PART NUMBER
DESCRIPTION
QTY ON HAND
UNIT COST

Program Output for Sample Control Break Program



```
07/05/83                    PHYSICAL INVENTORY REPORT                 PAGE   1

DEPT      PART                                 QUANTITY    UNIT       INVENTORY
 NO      NUMBER        DESCRIPTION             ON HAND     COST         VALUE

1000     110025        SOCKS                      100       3.00         300.00
1000     110723        JACKET                      30      20.00         600.00

                           TOTAL VALUE - DEPARTMENT 1000                $900.00*

2010   ,  22050        SOCKS                      100       1.25         125.00
2010      22444        SHIRT                       50      15.00         750.00

                           TOTAL VALUE - DEPARTMENT 2010                $875.00*

9415       9110        SHOES                       15      12.95         194.25
9415       9119        JACKETS                     10       9.95          99.50

                           TOTAL VALUE - DEPARTMENT 9415                $293.75*

                           FINAL TOTAL INVENTORY VALUE                $6,820.75**
```

IPO Chart and Hierarchy Chart for the Sample Program (Part 1 of 2)

Input/Processing/Output Chart

DATE: June 17

PROGRAMMER: John Doe

PROGRAM: Inventory Report

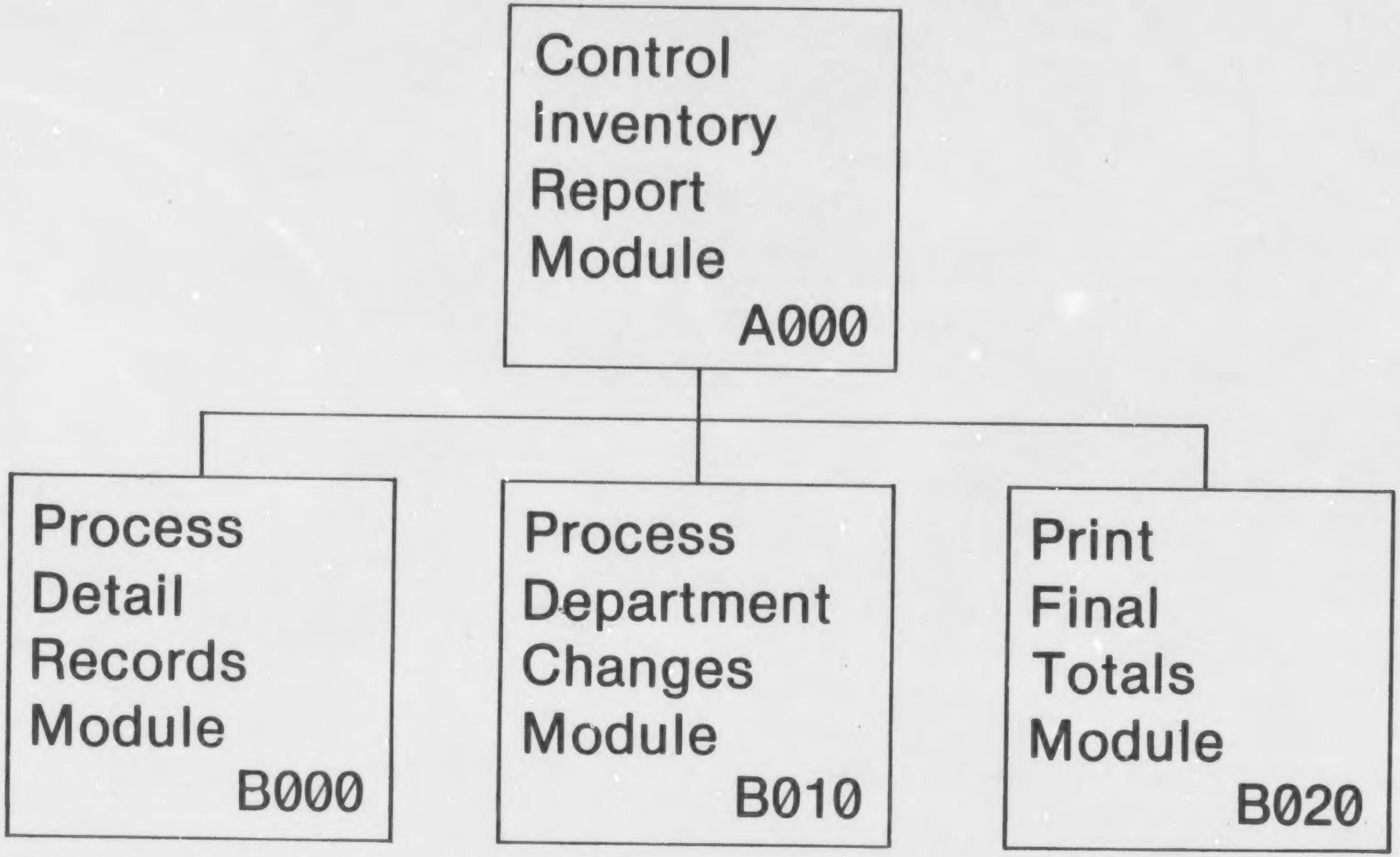
MODULE FUNCTION: Control Inventory Report

REF: A000

MODULE NAME: Control Inventory Report

INPUT	PROCESSING	OUTPUT
1. Inventory Input	1. Initialization	1. Physical Inventory
File	2. Obtain the input data	Report
	3. Process detail records	
	4. Process department changes	
	5. Print the final totals	
	6. Termination	

IPO Chart and Hierarchy Chart for the Sample Program (Part 2 of 2)



Control Inventory Report Module A000

Process Detail Records Module B000

Process Department Changes Module B010

Print Final Totals Module B020

IPO Chart for Process Detail Records Module

Input/Processing/Output Chart

DATE: June 17

PROGRAMMER: John Smith

PROGRAM: Inventory Report

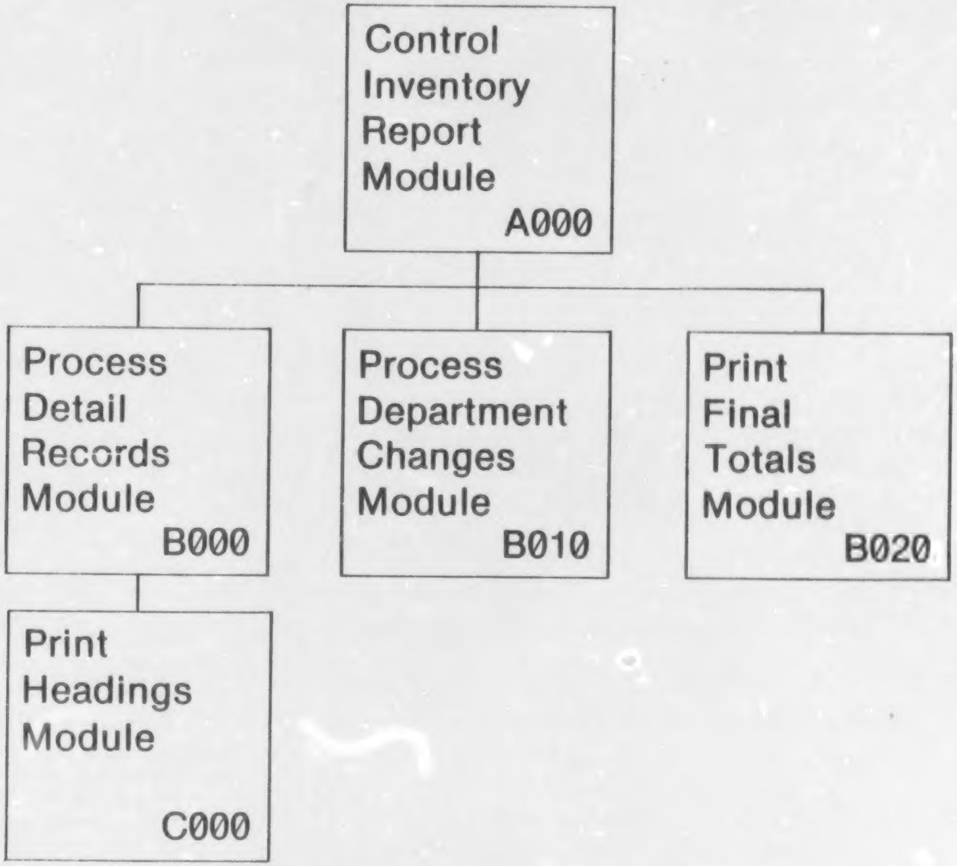
MODULE FUNCTION: Process the Detail Records

REF: B000

MODULE NAME: Process Detail Record

INPUT	PROCESSING	OUTPUT
1. Inventory Input	1. Print headings when required	1. Detail Print Line
Record	2. Format detail line	2. Updated Depart –
2. Department Total	3. Calculate inventory value	ment Total
Accumulator	4. Update accumulators	Accumulator
3. Final Total	a. Department totals	3. Updated Final
Accumulator	b. Final total	Total
	5. Write the detail line	Accumulator

Hierarchy Chart for Process Detail Records Module



Control Inventory Report Module A000

Process Detail Records Module B000

Process Department Changes Module B010

Print Final Totals Module B020

Print Headings Module C000

IPO Chart for Process Department Changes Module

Input/Processing/Output Chart

DATE: June 17	**PROGRAMMER:** John Smith	**PROGRAM:** Inventory Report
MODULE FUNCTION: Process Department Changes	**REF:** B010	**MODULE NAME:** Process Dept Changes

INPUT	PROCESSING	OUTPUT
1. Old Department	1. Format department total line	1. Department Total
Number	2. Write the department total line	Line
2. New Department	3. Reset department accumulator	2. Department Total
Number	4. Reset department number	Accumulator
3. Department Total	compare area	3. Department
Accumulator		Number
		Compare Area

IPO Chart for Print the Final Totals Module

Input/Processing/Output Chart

DATE: June 18	PROGRAMMER: Sandy Ash	PROGRAM: Inventory Report
MODULE FUNCTION: Print the Final Totals	REF: B020	MODULE NAME: Print Final Totals

INPUT	PROCESSING	OUTPUT
1. Final Total	1. Format total line	1. Final Total Line
Accumulator	2. Write the final total line	

TM 16

IPO Chart for Print the Headings Module

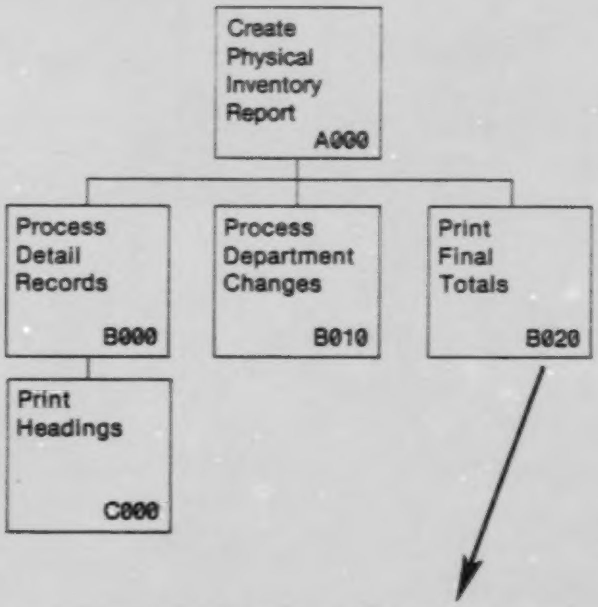
Input/Processing/Output Chart

DATE: June 18	PROGRAMMER: Ken Jones	PROGRAM: Inventory Report

MODULE FUNCTION: Print the Headings	REF: C000	MODULE NAME: Print Headings

INPUT	PROCESSING	OUTPUT
1. Heading Lines	1. Format headings	1. Heading Lines
	2. Write the headings	

Pseudocode Specifications for Final Total Printing


Create Physical Inventory Report A000
Process Detail Records B000
Process Department Changes B010
Print Final Totals B020
Print Headings C000

Input/Processing/Output Chart

DATE: June 18	PROGRAMMER: Sandy Ash		PROGRAM: Inventory Report
MODULE FUNCTION: Print the Final Totals		REF: B020	MODULE NAME: Print Final Totals

INPUT	PROCESSING	OUTPUT
1. Final Total	1. Format total line	1. Final Total Line
Accumulator	2. Write the final total line	

Pseudocode Specifications

DATE: June 20	PROGRAMMER: Sandy Ash		PROGRAM: Inventory Report
MODULE FUNCTION: Print the Final Totals		REF: B020	MODULE NAME: Print Final Totals

PSEUDOCODE	REF:	FILES, RECORDS, FIELDS REQUIRED
Move the final total value to the final total line		Final total accumulator
Write the final total line on the report		Final total line

Pseudocode for Printing the Headings

Pseudocode Specifications

DATE: June 20	**PROGRAMMER:** Ken Jones	**PROGRAM:** Physical Inventory Report
MODULE FUNCTION: Print the Headings	**REF:** C000	**MODULE NAME:** Print Headings

PSEUDOCODE	REF:	FILES, RECORDS, FIELDS REQUIRED
Move the current date to the first heading line Move the page number to the first heading line Write the first heading line Add 1 to the page count Write the second heading line Write the third heading line Set spacing for double spacing Reset lines printed counter to zero		Current date First heading line Page number (page count) Second heading line Third heading line Space control field Double spacing control character Number of lines printed counter

CONTROL BREAKS

UNIT VI

ASSIGNMENT SHEET 1: Terms and Background Information on Control Breaks

1. Briefly explain what a control break is.

2. Why is the input file to a control break program sorted into the same sequence as the desired control breaks?

3. A control break occurs when ____ .

CONTROL BREAKS

UNIT VI

ASSIGNMENT SHEET 2: Initial Detail and Second Detail Processing in Control Break Programs

1. Briefly explain the steps in initial processing in a control break program. (Refer to TM 4 and use you own words.)

2. Briefly explain the steps in the processing of first detail record in a control break program. (TM 5)

3. Briefly explain the steps in the processing of a detail record in a control break program. (TM 6)

CONTROL BREAKS

UNIT VI

ASSIGNMENT SHEET 3: Actual Control Break Processing and Final Total Control Break Processing

1. Briefly explain (in your own words) the steps in the actual processing of a control break. (TM 7)

2. Briefly explain the steps in final total processing. (TM 8)

3. Thought question: Why is a final total break like a control break?

CONTROL BREAKS

UNIT VI

ASSIGNMENT SHEET 4: A Sample Control Break Program

1. What is the name of the control break field in this application? (TM 9)

2. What is the name of the module which is the driver for the Inventory Report program?

3. Which two accumulators are updated by the Process Detail Record module?

4. Explain in your own words the function of the ENDIF and ENDPERFORM statements in the pseudocode for the Control Inventory Report Module. (TM 17)

5. Which module calculates the inventory value?

6. Why is the record key defined as COMP-3?

CONTROL BREAKS

UNIT VI

ANSWERS TO ASSIGNMENT SHEETS

Assignment Sheet 1:

1. A control break means that there has been a change in the control field or record key.
2. The sorting is necessary so that the program can output sequentially the records in correct order and so the program can sense a change in the control field.
3. The previous key field is not equal to the current key field.

Assignment Sheet 2:

1. a. Read the first record.
   b. Set up the Department Number in the compare area.
2. a. Compare Department Number in record to Department Number in compare area; if they are equal, this is not a control break.
   b. Print headings.
   c. Move data to output area.
   d. Calculate inventory value.
   e. Move inventory value to output.
   f. Accumulate Department Total and overall total.
   g. Print detail line.
3. a. Read record.
   b. Compare Department Number in record to Department Number in compare area; if they are equal, it is not a control break.
   c. Move input data to output.
   d. Calculate inventory value.
   e. Move inventory value to output.
   f. Accumulate Department and Final Totals.
   g. Print detail line.

Assignment Sheet 3:

1. a. Read record.
   b. Compare Department Numbers; if they are not equal it is a control break.
   c. Move Department Number from compare area to output.
   d. Move Department Total to output.
   e. Print the Department line.
   f. Zero Department Accumulator.
   g. Set up new Department Number.
2. a. Reader empty

198

   b. Print Department Number and total.
   c. Print Final Total.

3. A final total break is like a control break because the definition of a control break is that the record or control is not equal; this is true when the card reader is empty or the trailer record is read.

Assignment Sheet 4:

1. Department Number

2. Control Inventory Report Module

3. Department and Final Total

4. These indicate (for documentation purposes) the end of the IF statement and the end of the PERFORM's range. (They are not presently implemented in COBOL but are expected.)

5. Process Detail Records Module.

6. The record key is defined as COMP-3 because it is all numeric and internal processing; efficiencies are obtained.

199

JOB SHEET I: Writing a Payroll-Validation Report using Control Break Logic

Materials

This job is to be done in COBOL on your school's computer. As part of the assignment you will complete the documentation, code a COBOL solution, and test the solution. Coding forms should be used if available. Otherwise, use clean, lined notepaper.

Introduction

A simple but complete payroll validation report with a control break on department is the application for this job. In addition, you are provided with a similar sample application of an inventory report with a control break by branch location. The sample program will help you when designing and completing your payroll program, so study it carefully. At this time turn to the appendix and carefully review the sample programs, hierarchy chart, explanation, and listing.

Explanation of the Payroll Validation Report

A report should be prepared listing the amount to be paid for regular time and for overtime (1.5 times regular rate for over 40 hours) for all employees, grouped by department with a control break after each department. Input cards containing (for each employee) the hourly rate of pay and the number of hours worked are read in order by employee number **within department.** The input-card layout for the problem is given in Figure JS I-I.

No sample printer layout is given; each student doing the problem is required to design his or her own report format. The report should print the department name near or in the total line for that department. See sample program JS I-6 for example where branch location is similar to department.

Separate department totals should be kept for regular pay, overtime pay, and combined pay. A companywide total for each should be accumulated to print at the end of the report. The program should be coded with separate modules for regular-pay and overtime-pay calculations.

An input validation module should be included. Errors should be flagged with an appropriate message, and pay amounts for records in error should be excluded from the totals (but listed on the report). A sample control hierarchy chart for the payroll-validation program is given in Figure JS I-2.

Starting the Job

You will start the job by writing the procedure division for the payroll validation report. Note that the work is considerably lessened because you will write only the procedure division for this application. The other three divisions will be provided by your instructor. Figure JS I-3, Prewritten Divisions for Payroll Validation Report, is a printed copy of these first three divisions to be used in writing your procedure division.

At this time you should:

1. Design your output report format (using layout forms if available).
2. Draw a hierarchy chart for the application.

   HINT: use sample as a guide, but don't follow it too closely.
3. Write IPO charts for each module in your hierarchy chart (revise hierarchy chart if necessary).
4. Code each module.
5. Prepare test data and test your program.

Completing The Job

To complete this job, you should finish and hand in:

1. Layout of your report's design.
2. Hierarchy chart.
3. IPO charts.
4. Coding sheets.
5. Successful COBOL compilation and execution printout.

201

Figure JS1-1: Input Card Layout for Payroll Validation Report Program

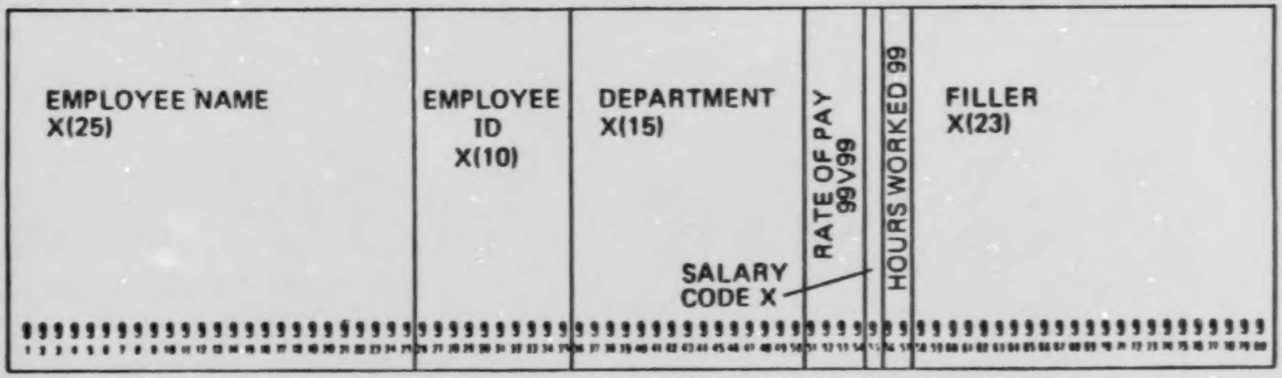
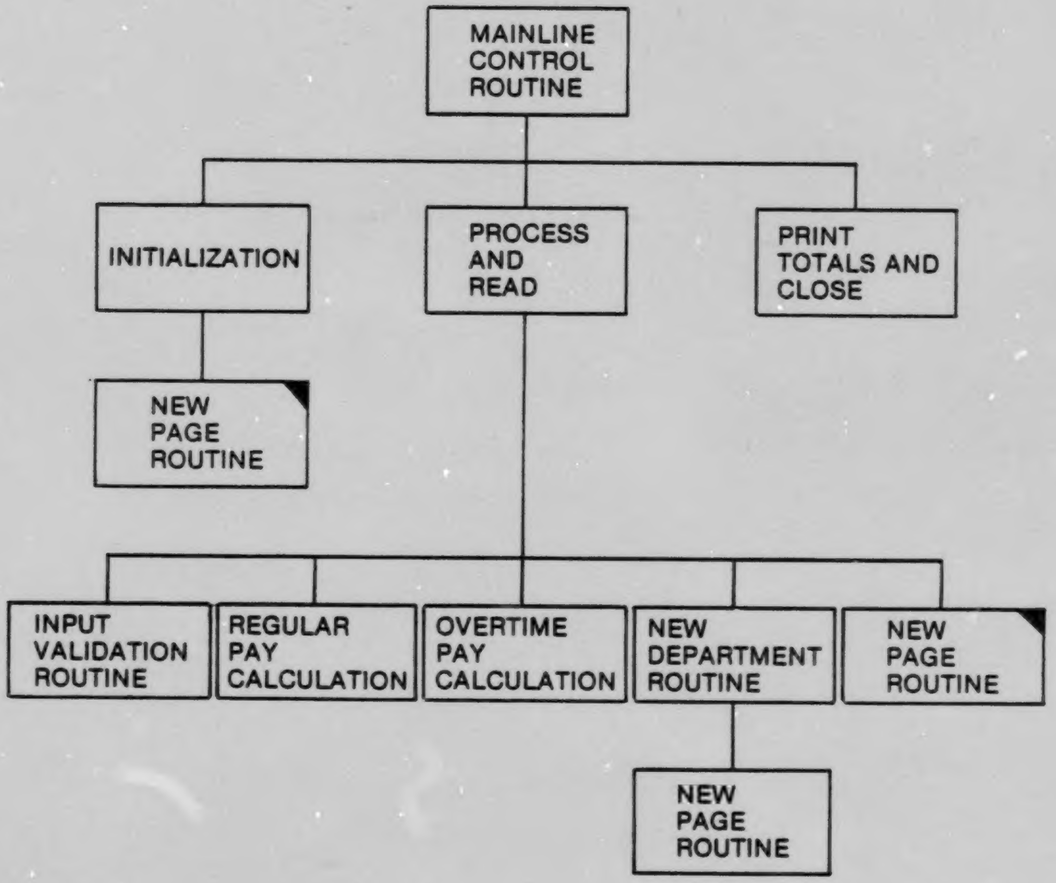

EMPLOYEE NAME
X(25)
EMPLOYEE
ID
X(10)
DEPARTMENT
X(15)
RATE OF PAY
99V99
SALARY
CODE X
HOURS WORKED 99
FILLER
X(23)

Figure JS1-2: Sample Control Hierarchy Chart for Payroll Validation Report Program


MAINLINE CONTROL ROUTINE
INITIALIZATION
PROCESS AND READ
PRINT TOTALS AND CLOSE
NEW PAGE ROUTINE
INPUT VALIDATION ROUTINE
REGULAR PAY CALCULATION
OVERTIME PAY CALCULATION
NEW DEPARTMENT ROUTINE
NEW PAGE ROUTINE
NEW PAGE ROUTINE

203

Figure JS1-3: Prewritten Divisions for Payroll Validation Report

```
IDENTIFICATION DIVISION.
PROGRAM-ID.  LISTER.

ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER.  IBM-370.
OBJECT-COMPUTER.  IBM-370.
SPECIAL-NAMES.
     C01 IS TOP-OF-PAGE.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
     SELECT PAYROLL-FILE ASSIGN TO UR-S-SYSIN.
     SELECT PAYROLL-LIST-FILE ASSIGN TO UR-S-SYSPRINT.
DATA DIVISION.
FILE SECTION.
FD  PAYROLL-FILE
    LABEL RECORDS ARE OMITTED.
01  PAYROLL-RECORD                      PIC X(80).
FD  PAYROLL-LIST-FILE
    LABEL RECORDS ARE OMITTED.
01  PRINT-LINE                          PIC X(121).
WORKING-STORAGE SECTION.
01  PAYROLL-DATA.
    05  EMPLOYEE-NAME                   PIC X(25).
    05  EMPLOYEE-ID                     PIC X(10).
    05  DEPARTMENT                      PIC X(15).
    05  RATE-OF-PAY                     PIC 99V99.
    05  SALARY-CODE                     PIC X.
    05  HOURS-WORKED                    PIC 99.
    05  FILLER                          PIC X(23).
01  DETAIL-LINE.
    05  FILLER                          PIC X(2).
    05  D-EMPLOYEE-NAME                 PIC X(25).
    05  FILLER                          PIC X(2).
    05  D-EMPLOYEE-ID                   PIC X(10).
    05  FILLER                          PIC X(3).
    05  D-RATE-OF-PAY                   PIC $Z9.99.
    05  FILLER                          PIC X(5).
    05  D-HOURS-WORKED                  PIC Z9.
    05  FILLER                          PIC X(4).
    05  D-REGULAR                       PIC $ZZZ9.99.
    05  FILLER                          PIC X(5).
    05  D-OVERTIME                      PIC $ZZZ9.99.
    05  FILLER                          PIC X(5).
    05  D-TOTAL-PAY                     PIC $ZZZZ9.99.
    05  FILLER                          PIC X(5).
    05  ERROR-MESSAGE                   PIC X(35).

01  REPORT-TITLE.
    05  FILLER                          PIC X(50) VALUE SPACES
    05  FILLER                          PIC X(19) VALUE
                       'PAY RATE REPORT'.
    05  FILLER                          PIC X(53) VALUE SPACES
01  COLUMN-HEADER.
    05  FILLER                          PIC X(2) VALUE SPACES.
    05  FILLER                          PIC X(13) VALUE
                       'EMPLOYEE NAME'.
    05  FILLER                          PIC X(14) VALUE SPACES.
    05  FILLER                          PIC X(9)  VALUE
                       'ID NUMBER'.
    05  FILLER                          PIC X(3) VALUE SPACES.
    05  FILLER                          PIC X(8)  VALUE
                       'PAY RATE'.
    05  FILLER                          PIC X(2)  VALUE SPACES.
    05  FILLER                          PIC X(5)  VALUE 'HOURS'.
    05  FILLER                          PIC X(4)  VALUE SPACES.
    05  FILLER                         PIC X(7) VALUE 'REGULAR'.
    05  FILLER                          PIC X(5)  VALUE SPACES.
    05  FILLER                          PIC X(8)  VALUE
                       'OVERTIME'.
    05  FILLER                          PIC X(5)  VALUE SPACES.
    05  FILLER                          PIC X(9)  VALUE
                       'TOTAL PAY'.
    05  FILLER                          PIC X(5)  VALUE SPACES.
    05  FILLER                          PIC X(13) VALUE
                       'ERROR MESSAGE'.
01  TOTAL-LINE.
    05  FILLER                          PIC X(2) VALUE SPACES.
    05  FILLER                          PIC X(10) VALUE
                          'TOTAL FOR '.
    05  D-EMP-DEPT-A                    PIC X(14).
    05  FILLER                         PIC X(7) VALUE ' EQUALS'.
    05  FILLER                          PIC X(25) VALUE SPACES.
    05  T-REGULAR                       PIC $ZZZZ9.99.
    05  FILLER                          PIC X(4)  VALUE SPACES.
    05  T-OVERTIME                      PIC $ZZZZ9.99.
    05  FILLER                          PIC X(4)  VALUE SPACES.
    05  T-TOTAL-PAY                     PIC $ZZZZZ9.99.
01  END-TOTAL-LINE.
    05  FILLER                          PIC X(2)  VALUE SPACES.
    05  FILLER                          PIC X(26) VALUE
                          'TOTALS FOR THE COMPANY ARE'.
    05  FILLER                          PIC X(30) VALUE SPACES.
    05  E-C-REG                         PIC $ZZZZ9.99.
    05  FILLER                          PIC X(4)  VALUE SPACES.

    05  E-C-OVER                        PIC $ZZZZ9.99.
    05  FILLER                          PIC X(4)  VALUE SPACES.
    05  E-C-TOTAL                       PIC $ZZZZZ9.99.
01  BLANK-LINE.
    05  FILLER                        PIC X(121) VALUE SPACES.
01  END-LINE.
    05  FILLER                          PIC X(2)  VALUE SPACES.
    05  FILLER                          PIC X(14) VALUE
                          'END OF REPORT'.
01  DEPT-LINE.
    05  FILLER                          PIC X(2)  VALUE SPACES.
    05  D-EMP-DEPT                      PIC X(15) VALUE SPACES.
    05  FILLER                          PIC X(105) VALUE SPACES.
01  C-TOTAL                           PIC 99999V99  VALUE ZERO.
01  C-REG                             PIC 99999V99  VALUE ZERO.
01  C-OVER                            PIC 99999V99  VALUE ZERO.
```

```
01  TOTAL-DEPT                        PIC 9999V99  VALUE ZERO.
01  OVER-DEPT                         PIC 9999V99  VALUE ZERO.
01  REG-DEPT                          PIC 9999V99  VALUE ZERO.
01  REGULAR                           PIC 9999V99  VALUE ZERO.
01  OVERTIME                          PIC 9999V99  VALUE ZERO.
01  TOTAL-PAY                         PIC 9999V99  VALUE ZERO.
01  LINE-COUNT                        PIC 99       VALUE ZERO.
01  LINE-LIMIT                        PIC 99        VALUE 55.
01  OVERTIME-HOUR                     PIC 99       VALUE ZERO.
01  REGULAR-HOURS                     PIC 99       VALUE ZERO.
01  PREVIOUS-EMP-DEPT                 PIC X(14)  VALUE SPACES.
01  O-T-RATE                          PIC 99V99  VALUE ZERO.
01  FORTY                             PIC 99     VALUE 40.
01  ERROR-SWITCH                      PIC X(1)   VALUE 'N'.
    88 THERE-IS-AN-ERROR                         VALUE 'Y'.
    88 THERE-IS-NOT-AN-ERROR                     VALUE 'N'.
01  FIRST-TIME-THRU-SWITCH            PIC X(1)   VALUE 'Y'.
    88 FIRST-TIME-THRU                           VALUE 'Y'.
    88 NOT-FIRST-TIME-THRU                       VALUE 'N'.
01  MORE-DATA                         PIC X      VALUE 'Y'.
    88  THERE-IS-MORE-DATA                       VALUE 'Y'.
    88  THERE-IS-NO-MORE-DATA                    VALUE 'N'.
```

Figure JS1-3: Prewritten Divisions for Payroll Validation Report

```
IDENTIFICATION DIVISION.
PROGRAM-ID.  LISTER.

ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER.  IBM-370.
OBJECT-COMPUTER.  IBM-370.
SPECIAL-NAMES.
        C01 IS TOP-OF-PAGE.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
        SELECT PAYROLL-FILE ASSIGN TO UR-S-SYSIN.
        SELECT PAYROLL-LIST-FILE ASSIGN TO UR-S-SYSPRINT.
DATA DIVISION.
FILE SECTION.
FD  PAYROLL-FILE
    LABEL RECORDS ARE OMITTED.
01  PAYROLL-RECORD                  PIC X(80).
FD  PAYROLL-LIST-FILE
    LABEL RECORDS ARE OMITTED.
01  PRINT-LINE                      PIC X(121).
WORKING-STORAGE SECTION.
01  PAYROLL-DATA.
    05  EMPLOYEE-NAME               PIC X(25).
    05  EMPLOYEE-ID                 PIC X(10).
    05  DEPARTMENT                  PIC X(15).
    05  RATE-OF-PAY                 PIC 99V99.
    05  SALARY-CODE                 PIC X.
    05  HOURS-WORKED                PIC 99.
    05  FILLER                      PIC X(23).
01  DETAIL-LINE.
    05  FILLER                      PIC X(2).
    05  D-EMPLOYEE-NAME             PIC X(25).
    05  FILLER                      PIC X(2).
    05  D-EMPLOYEE-ID               PIC X(10).
    05  FILLER                      PIC X(3).
    05  D-RATE-OF-PAY               PIC $Z9.99.
    05  FILLER                      PIC X(5).
    05  D-HOURS-WORKED              PIC Z9.
    05  FILLER                      PIC X(4).
    05  D-REGULAR                   PIC $ZZZ9.99.
    05  FILLER                      PIC X(5).
    05  D-OVERTIME                  PIC $ZZZ9.99.
    05  FILLER                      PIC X(5).
    05  D-TOTAL-PAY                 PIC $ZZZZ9.99.
    05  FILLER                      PIC X(5).
    05  ERROR-MESSAGE               PIC X(35).

01  REPORT-TITLE.
    05  FILLER                      PIC X(50) VALUE SPACES
    05  FILLER                      PIC X(19) VALUE
                     'PAY RATE REPORT'.
    05  FILLER                      PIC X(53) VALUE SPACES
01  COLUMN-HEADER.
    05  FILLER                      PIC X(2) VALUE SPACES.
    05  FILLER                      PIC X(13) VALUE
                          'EMPLOYEE NAME'.
    05  FILLER                           PIC X(14) VALUE SPACES.
    05  FILLER                           PIC X(9)  VALUE
                          'ID NUMBER'.
    05  FILLER                           PIC X(3) VALUE SPACES.
    05  FILLER                           PIC X(8)  VALUE
                          'PAY RATE'.
    05  FILLER                           PIC X(2)  VALUE SPACES.
    05  FILLER                           PIC X(5)  VALUE 'HOURS'.
    05  FILLER                           PIC X(4)  VALUE SPACES.
    05  FILLER                          PIC X(7) VALUE 'REGULAR'.
    05  FILLER                           PIC X(5)  VALUE SPACES.
    05  FILLER                           PIC X(8)  VALUE
                          'OVERTIME'.
    05  FILLER                           PIC X(5)  VALUE SPACES.
    05  FILLER                           PIC X(9)  VALUE
                          'TOTAL PAY'.
    05  FILLER                           PIC X(5)  VALUE SPACES.
    05  FILLER                           PIC X(13) VALUE
                          'ERROR MESSAGE'.
01  TOTAL-LINE.
    05  FILLER                           PIC X(2) VALUE SPACES.
    05  FILLER                           PIC X(10) VALUE
                             'TOTAL FOR '.
    05  D-EMP-DEPT-A                     PIC X(14).
    05  FILLER                          PIC X(7) VALUE ' EQUALS'.
    05  FILLER                           PIC X(25) VALUE SPACES.
    05  T-REGULAR                        PIC $ZZZZ9.99.
    05  FILLER                           PIC X(4)  VALUE SPACES.
    05  T-OVERTIME                       PIC $ZZZZ9.99.
    05  FILLER                           PIC X(4)  VALUE SPACES.
    05  T-TOTAL-PAY                      PIC $ZZZZZ9.99.
01  END-TOTAL-LINE.
    05  FILLER                           PIC X(2)  VALUE SPACES.
    05  FILLER                           PIC X(26) VALUE
                             'TOTALS FOR THE COMPANY ARE'.
    05  FILLER                           PIC X(30) VALUE SPACES.
    05  E-C-REG                          PIC $ZZZZ9.99.
    05  FILLER                           PIC X(4)  VALUE SPACES.

    05  E-C-OVER                         PIC $ZZZZ9.99.
    05  FILLER                           PIC X(4)  VALUE SPACES.
    05  E-C-TOTAL                        PIC $ZZZZZ9.99.
01  BLANK-LINE.
    05  FILLER                          PIC X(121) VALUE SPACES.
01  END-LINE.
    05  FILLER                           PIC X(2)  VALUE SPACES.
    05  FILLER                           PIC X(14) VALUE
                             'END OF REPORT'.
01  DEPT-LINE.
    05  FILLER                           PIC X(2)  VALUE SPACES.
    05  D-EMP-DEPT                       PIC X(15) VALUE SPACES.
    05  FILLER                           PIC X(105) VALUE SPACES.
01  C-TOTAL                             PIC 99999V99  VALUE ZERO.
01  C-REG                               PIC 99999V99  VALUE ZERO.
01  C-OVER                              PIC 99999V99  VALUE ZERO.
```

```
01  TOTAL-DEPT                          PIC 9999V99  VALUE ZERO.
01  OVER-DEPT                          PIC 9999V99   VALUE ZERO.
01  REG-DEPT                           PIC 9999V99   VALUE ZERO.
01  REGULAR                            PIC 9999V99   VALUE ZERO.
01  OVERTIME                           PIC 9999V99   VALUE ZERO.
01  TOTAL-PAY                          PIC 9999V99   VALUE ZERO.
01  LINE-COUNT                         PIC 99        VALUE ZERO.
01  LINE-LIMIT                          PIC 99        VALUE 55.
01  OVERTIME-HOUR                      PIC 99        VALUE ZERO.
01  REGULAR-HOURS                       PIC 99       VALUE ZERO.
01  PREVIOUS-EMP-DEPT                   PIC X(14)  VALUE SPACES.
01  O-T-RATE                            PIC 99V99   VALUE ZERO.
01  FORTY                               PIC 99      VALUE 40.
01  ERROR-SWITCH                        PIC X(1)    VALUE 'N'.
    88 THERE-IS-AN-ERROR                            VALUE 'Y'.
    88 THERE-IS-NOT-AN-ERROR                        VALUE 'N'.
01  FIRST-TIME-THRU-SWITCH              PIC X(1)    VALUE 'Y'.
    88 FIRST-TIME-THRU                              VALUE 'Y'.
    88 NOT-FIRST-TIME-THRU                          VALUE 'N'.
01  MORE-DATA                           PIC X       VALUE 'Y'.
    88  THERE-IS-MORE-DATA                          VALUE 'Y'.
    88  THERE-IS-NO-MORE-DATA                       VALUE 'N'.
```

APPENDIX

JOB SHEET 1: Sample Application, an Inventory List by Branch Location Report Application

The sample program in this appendix prints a report that lists parts grouped by branch location. Each entry shows the part, description, price, quantity on hand, and total value. For the total cost of the inventory is printed.

The example card layout for the program is shown in JS 1-4. The input-data will be arranged by part number within location.

A hierarchy chart for the example program is given in JS1-5, and a complete program listing is shown in JS 1-6. Note, in the hierarchy chart, that the module NEW-PAGE-ROUTINE is shown four times. This repetition indicates that in the program the module is called (PERFORMed) from four different places in the program. Ordinarily, one box can be designated as the major use of the module. The convention used in this text to indicate nonmajor use of modules will be to place a slash in the upper right corner of a box to indicate that the major use of the module is elsewhere. In this case (JS 1-5), the module OLD-LOCATION-ROUTINE is normally called (PERFORMed) by LOCATION-BREAK-ROUTINE. It is also performed by the END-ROUTINE. Therefore the box is repeated under that module with a slash in the upper right corner.

The program should be studied carefully and referred to while reading the following discussion of the features it incorporates.

A sample report output of the program is shown in JS 1-7. For each branch location, the branch number prints on a separate line before the list of parts. At the end of the list for that branch, a separate total line has the label "LOCATION SUBTOTAL" and the total amount for the parts on hand for that branch location.

When a list of parts for a branch runs over two pages, the branch number is repeated at the top of the new page.

The program contains examples of several features introduced in Chapter 4A, COMPUTE and nested IF statements. There is an input-validation module, which tests each record and prints an error message for each record in error. The error message indicates the reason for the first error detected. If an error is located, no further tests are made for more errors. The erroneous record is rejected, and the dollar amount for that record is not included in the total for the publisher order.

The program illustrates the concept of a **control break.** Each time a record is read, its branch location value is compared to the branch location value for the record just listed. If they are different, a module called LOCATION-BREAK-ROUTINE is performed. This module formats and prints totals for the branch location just completed, skips several lines, and prints the number of the next location. Control is then returned to processing of the next detail line.

The program keeps a line count, and a module called NEW-PAGE-ROUTINE is invoked when the maximum line count is reached. If a maximum line count and a branch location break occur at the same time, the totals for the finished branch location are printed at the end of the page. NEW-PAGE-ROUTINE is then performed, and the new branch location name prints on the next page.

Not only is the total dollar order for each branch printed at the end of each branch listing (at each control break), but at the end of the program, a total dollar amount of all inventory is printed.

In summary, the branch location inventory program should be examined for its use of:

1. COMPUTE
2. Nested IFs
3. Input validation
4. Use of control breaks and totals at each break
5. Control over lines on each page
6. Headings on the new page
7. Management of correct processing order for control break and new page

Figure JS1-4: Input Layout for Inventory by Branch Location

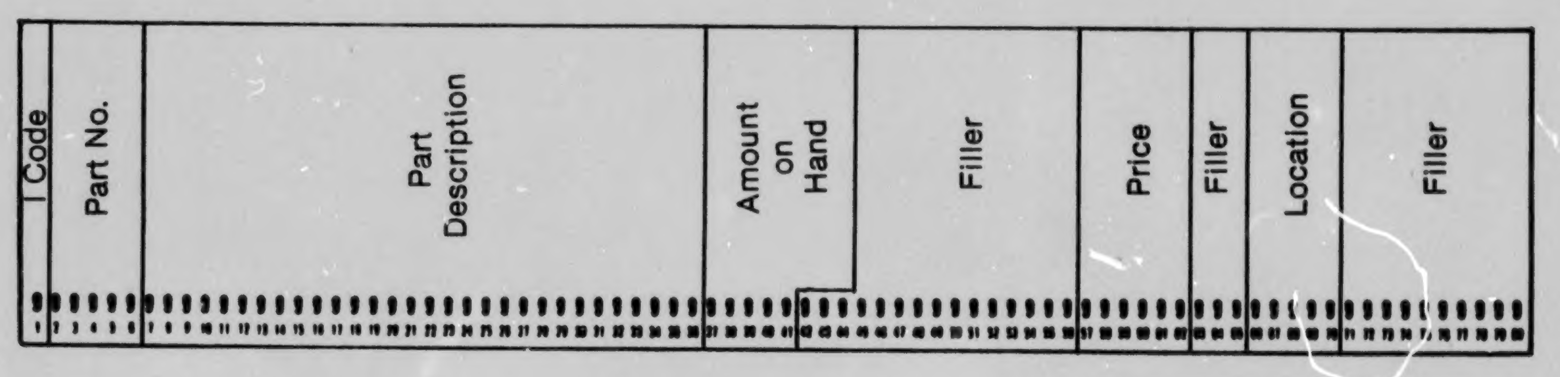

I Code
Part No.
Part Description
Amount on Hand
Filler
Price
Filler
Location
Filler

Figure JS1-5 Hierarchy Chart for Inventory-Report by Branch-Location Program

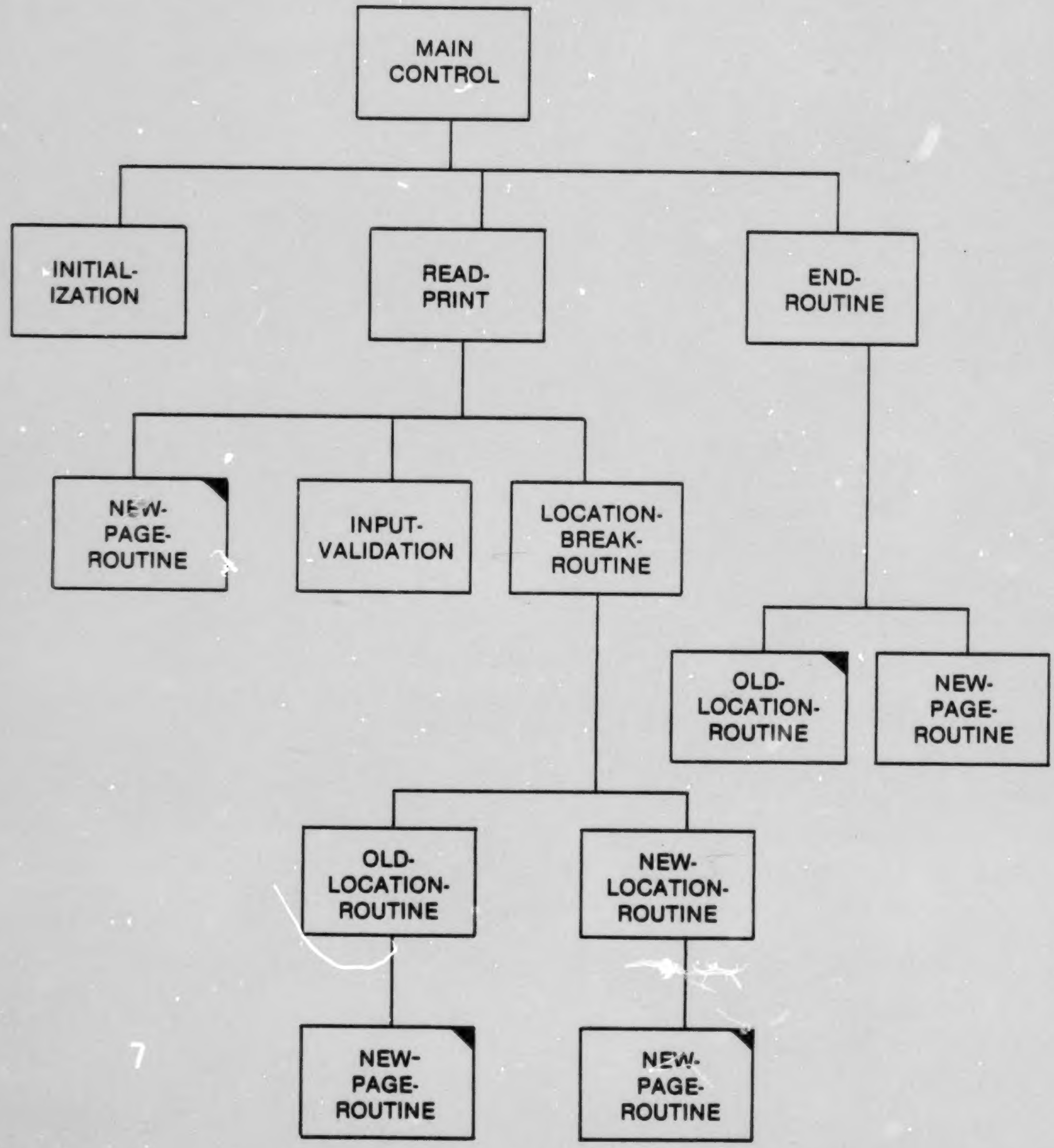

MAIN CONTROL
INITIAL-IZATION
READ-PRINT
END-ROUTINE
NEW-PAGE-ROUTINE
INPUT-VALIDATION
LOCATION-BREAK-ROUTINE
OLD-LOCATION-ROUTINE
NEW-PAGE-ROUTINE
OLD-LOCATION-ROUTINE
NEW-LOCATION-ROUTINE
NEW-PAGE-ROUTINE
NEW-PAGE-ROUTINE

Figure JS1-6: Program Listing of Sample Application for Inventory by Branch (Part 1 of 2)

```
     IDENTIFICATION DIVISION.
     PROGRAM-ID. INVLIST.
     AUTHOR. CHUCK LITECKY.
     REMARKS. PROGRAM      DL04B COBOL ON ACCT C1136
              WRITTEN FOR DLO, 'ELEM. COBOL PROGRAMMING'
                 CHAPTER 4B.
     ENVIRONMENT DIVISION.
     CONFIGURATION SECTION.
     SOURCE-COMPUTER. IBM-370.
     OBJECT-COMPUTER. IBM-370.
     SPECIAL-NAMES.
          C01 IS TO-TOP-OF-PAGE.
     INPUT-OUTPUT SECTION.
     FILE-CONTROL.
          SELECT INVENTORY-FILE ASSIGN TO UR-S-SYSIN.
          SELECT PRINT-FILE ASSIGN TO UR-S-SYSPRINT.
     DATA DIVISION.
     FD  INVENTORY-FILE
         LABEL RECORDS ARE OMITTED.
     01  INVENTORY-RECORD           PIC X(80).
     FD  PRINT-FILE
         LABEL RECORDS ARE OMITTED.
     01  PRINT-LINE                 PIC X(132).
     WORKING-STORAGE SECTION.
1    01  MORE-DATA                  PIC X  VALUE 'Y'.
         88  THERE-IS-MORE-DATA            VALUE 'Y'.
         88  THERE-IS-NO-MORE-DATA         VALUE 'N'.
2    01  FIRST-TIME-THRU-SWITCH     PIC X(01) VALUE 'Y'.
         88  FIRST-TIME-THRU                  VALUE 'Y'.
         88  NOT-FIRST-TIME-THRU              VALUE 'N'.
3    01  PREVIOUS-LOCATION          PIC X(05) VALUE SPACE.
4    01  LINE-LIMIT                 PIC 9(02) VALUE 35.
5    01  ERROR-SWITCH               PIC X(01) VALUE 'N'.
         88  THERE-IS-AN-ERROR                VALUE 'Y'.
         88  NOT-AN-ERROR                     VALUE 'N'.
6    01  LINE-COUNT                 PIC 9(02) VALUE 55.
7    01  TOTAL-INVEN                PIC 9(7)V99 VALUE ZERO.
8    01  PART-TOTAL                 PIC 9(6)V99 VALUE ZERO.
9    01  LOCATION-TOTAL             PIC 9(6)V99 VALUE ZERO.
     01  INVENTORY-DATA.
         05  INVENTORY-CODE         PIC X(01).
         05  PART-NUM               PIC X(05).
         05  DESCRIPTION            PIC X(30).
         05  AMT-ON-HAND            PIC 9(05).
         05  FILLER                 PIC X(15).
         05  PRICE                  PIC 9(04)V99.
         05  FILLER                 PIC X(03).
         05  LOCATION               PIC X(05).
         05  FILLER                 PIC X(10).
```

```
        01  DETAIL-LINE.
            05  FILLER                  PIC X(08).
            05  D-PART-NUM              PIC X(05).
            05  FILLER                  PIC X(05).
            05  D-DESCRIPTION           PIC X(30).
            05  FILLER                  PIC X(02).
            05  D-AMT-ON-HAND           PIC ZZZZ9.
            05  FILLER                  PIC X(05).
            05  D-PRICE                 PIC $$$$9.99.
            05  FILLER                  PIC X(02).
          05  ERROR-MESSAGE           PIC X(20).
            05  FILLER                  PIC X(52).
        01  REPORT-TITLE.
10          05  FILLER                  PIC X(17) VALUE SPACES.
11          05  FILLER                  PIC X(51) VALUE
                'B R A N C H   I N V E N T O R Y   R E P O R T'.
12          05  FILLER                  PIC X(64) VALUE SPACES.
        01  COLUMN-HEADER.
13          05  FILLER                  PIC X(01) VALUE SPACE.
14          05  FILLER                  PIC X(06) VALUE 'BRANCH'.
15          05  FILLER                  PIC X(02) VALUE SPACE.
16          05  FILLER                  PIC X(08) VALUE 'PART NO.'.
17          05  FILLER                  PIC X(05) VALUE SPACES.
18          05  FILLER                  PIC X(23) VALUE
                'PART DESCRIPTION       '.
19          05  FILLER                  PIC X(14) VALUE 'AMOUNT ON HAND'.
20          05  FILLER                  PIC X(03) VALUE SPACES.
21          05  FILLER                  PIC X(05) VALUE 'PRICE'.
22          05  FILLER                  PIC X(02) VALUE SPACE.
23          05  FILLER                  PIC X(05) VALUE 'ERROR'.
24          05  FILLER                  PIC X(57) VALUE SPACE.
        01  LOCATION-HEADER.
25          05  LH-LOCATION             PIC X(05) VALUE SPACE.
26          05  FILLER                  PIC X(127) VALUE SPACE.
        01  TOTAL-LINE.
27          05  FILLER                  PIC X(31) VALUE SPACES.
28          05  FILLER                  PIC X(24) VALUE
                'TOTAL VALUE OF PARTS IS '.
            05  T-TOTAL-INVEN           PIC $Z,ZZZ,ZZ9.99.
29          05  FILLER                  PIC X(53) VALUE SPACES.
        01  LOCATION-SUBTOTAL-LINE.
30          05  FILLER                  PIC X(31) VALUE SPACE.
31          05  FILLER                  PIC X(24) VALUE
                'LOCATION SUBTOTAL IS    '.
            05  L-TOTAL-INVEN           PIC $Z,ZZZ,ZZ9.99.
32          05  FILLER                  PIC X(53) VALUE SPACE.
33      PROCEDURE DIVISION.
        MAIN-CONTROL.
34          PERFORM INITIALIZATION.
```

Figure JS1-6: Program Listing of Sample Application for Inventory by Branch (Part 2 of 2)

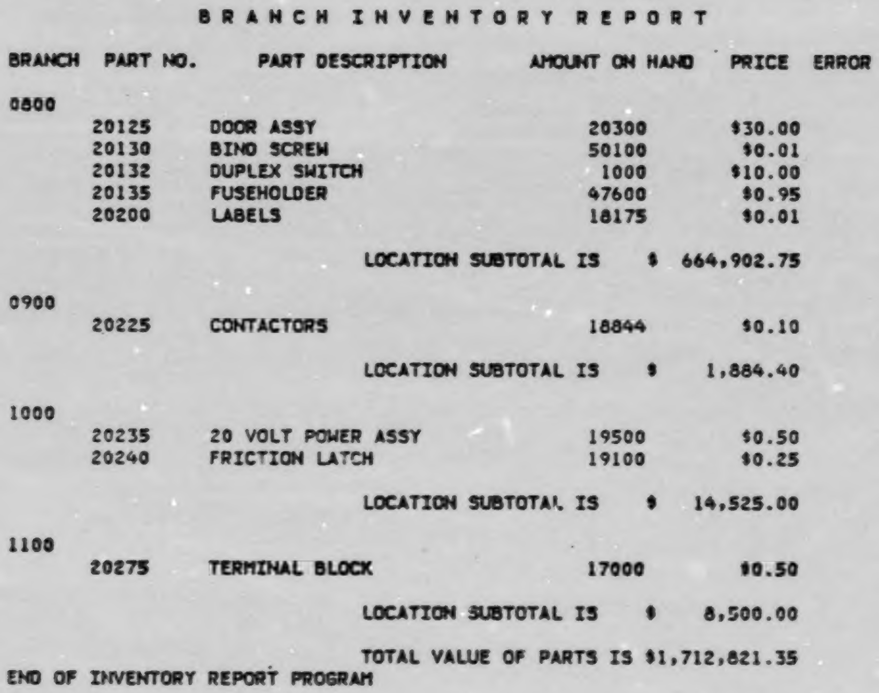
```
    PERFORM READ-PRINT UNTIL THERE-IS-NO-MORE-DATA.
    PERFORM END-ROUTINE.
    STOP RUN.
 INITIALIZATION.
    OPEN INPUT INVENTORY-FILE
         OUTPUT PRINT-FILE.
     MOVE SPACES TO DETAIL-LINE.
 READ-PRINT.
    READ INVENTORY-FILE INTO INVENTORY-DATA
          AT END MOVE 'N' TO MORE-DATA.
    IF THERE-IS-MORE-DATA
         PERFORM INPUT-VALIDATION
         IF THERE-IS-AN-ERROR
             NEXT SENTENCE
         ELSE
             IF PREVIOUS-LOCATION NOT EQUAL LOCATION
                 MOVE LOCATION TO PREVIOUS-LOCATION
                 PERFORM LOCATION-BREAK-ROUTINE.
**   A TRAILER RECORD WOULD PREVENT THIS CLUMSINESS AS ALSO
**      WOULD THE PERFORM WHILE.
    IF THERE-IS-MORE-DATA
         IF NOT-AN-ERROR
             COMPUTE PART-TOTAL = AMT-ON-HAND * PRICE
             COMPUTE LOCATION-TOTAL = LOCATION-TOTAL +
                 PART-TOTAL
             COMPUTE TOTAL-INVEN= TOTAL-INVEN + PART-TOTAL.

    IF THERE-IS-MORE-DATA
         MOVE PART-NUM TO D-PART-NUM
         MOVE DESCRIPTION TO D-DESCRIPTION
         MOVE AMT-ON-HAND TO D-AMT-ON-HAND
         MOVE PRICE TO D-PRICE
         IF LINE-COUNT IS GREATER THAN LINE-LIMIT
            PERFORM NEW-PAGE-ROUTINE.
     IF THERE-IS-MORE-DATA
         WRITE PRINT-LINE FROM DETAIL-LINE
             AFTER ADVANCING 1 LINES.
  INPUT-VALIDATION.
     MOVE SPACE TO ERROR-MESSAGE.
     MOVE 'N' TO ERROR-SWITCH.
     IF INVENTORY-CODE NOT EQUAL 'I'
        MOVE 'Y' TO ERROR-SWITCH
        MOVE 'NOT INVENTORY' TO ERROR-MESSAGE
     ELSE
         IF PART-NUM EQUAL SPACES
            MOVE 'Y' TO ERROR-SWITCH
            MOVE 'MISSING PART NO.' TO ERROR-MESSAGE
         ELSE
             IF DESCRIPTION EQUAL SPACES
                MOVE 'Y' TO ERROR-SWITCH
                MOVE 'MISSING DECCRIPTION' TO ERROR-MESSAGE
             ELSE
               IF AMT-ON-HAND NOT NUMERIC
                            OR
                    AMT-ON-HAND EQUAL ZERO
                    MOVE 'Y' TO ERROR-SWITCH
                    MOVE 'INVALID AMOUNT' TO ERROR-MESSAGE
                 ELSE
                     IF PRICE NOT NUMERIC
                               OR
                        PRICE EQUAL ZERO
                        MOVE 'Y' TO ERROR-SWITCH
                        MOVE 'INVALID PRICE' TO
                            ERROR-MESSAGE
                     ELSE
                         IF LOCATION EQUAL SPACE
                             MOVE 'Y' TO ERROR-SWITCH
                             MOVE 'MISSING LOCATION'
                                 TO ERROR-MESSAGE.
 LOCATION-BREAK-ROUTINE.
     IF FIRST-TIME-THRU
         MOVE 'N' TO FIRST-TIME-THRU-SWITCH
         PERFORM NEW-LOCATION-ROUTINE
     ELSE
        PERFORM OLD-LOCATION-ROUTINE
        PERFORM NEW-LOCATION-ROUTINE.
 OLD-LOCATION-ROUTINE.
*      ******   SETUP LOCATION FOOTING LINE AND PRINT  *****
      MOVE LOCATION-TOTAL TO L-TOTAL-INVEN.
         WRITE PRINT-LINE FROM LOCATION-SUBTOTAL-LINE
             AFTER ADVANCING 2 LINES.
      ADD 2 TO LINE-COUNT.
      MOVE ZERO TO LOCATION-TOTAL.
 NEW-LOCATION-ROUTINE.
*          ******   SETUP NEW LOCATION HEADER   AND PRINT *****
      MOVE LOCATION TO LH-LOCATION.
      IF LINE-COUNT GREATER THAN LINE-LIMIT
          PERFORM NEW-PAGE-ROUTINE.
      WRITE PRINT-LINE FROM LOCATION-HEADER
          AFTER ADVANCING 2 LINES
      ADD 2 TO LINE-COUNT.
  NEW-PAGE-ROUTINE.
      WRITE PRINT-LINE FROM REPORT-TITLE
          AFTER ADVANCING TO-TOP-OF-PAGE.
      WRITE PRINT-LINE FROM COLUMN-HEADER
          AFTER ADVANCING 2 LINES.
      MOVE 5 TO LINE-COUNT.
  END-ROUTINE.
      PERFORM OLD-LOCATION-ROUTINE.
       MOVE TOTAL-INVEN TO T-TOTAL-INVEN.
       WRITE PRINT-LINE FROM TOTAL-LINE
            AFTER ADVANCING 2 LINES.
       DISPLAY 'END OF INVENTORY REPORT PROGRAM'.
       CLOSE INVENTORY-FILE
             PRINT-FILE.

***** THERE ARE NO STATEMENTS FLAGGED IN THIS COMPILE
```

Figure JS1-7: Sample Output of Inventory Listing by Branch Location (Part 1 of 2)

```
               B R A N C H   I N V E N T O R Y   R E P O R T

BRANCH  PART NO.    PART DESCRIPTION      AMOUNT ON HAND    PRICE  ERROR

0121
       19010    TORSION BAR                    25000        $0.35
       19030    STRIKER                        60000        $0.02
       19045    EXTENSION SPRING               99900        $0.10
       20005    COMPRESSION SPRING              7517        $1.00

                             LOCATION SUBTOTAL IS   $   27,457.00

0201
       20010    STACKER                        31000        $0.01
       20015    TOGGLE BOLT                        0        $0.00  INVALID AMOUNT
       20019    TOGGLE BOLT EXTENDER             145        $0.00  INVALID PRICE
       20025    SCREWS                         59900        $0.03
       20045    TOGGLE SWITCH                  78000        $2.00
       20095    ROTARY BLADE                     100       $15.00

                             LOCATION SUBTOTAL IS   $  159,607.00

0222
       20115    HANDLES                        17600        $3.50
       20220    BUTTONS                        44400        $0.40

                             LOCATION SUBTOTAL IS   $   79,360.00

0233
       20125    DOOR ASSY                      20300       $30.00
       20130    BIND SCREW                     50100        $0.01
       20132    DUPLEX SWITCH                   1000       $10.00

                             LOCATION SUBTOTAL IS   $  619,501.00

0300
       20135    FUSEHOLDER                     47600        $0.95
       20145    LAG FUSE                       20400        $0.16

                             LOCATION SUBTOTAL IS   $   40,484.00

0500
       20165    HEX NUT                        17800        $0.05

                             LOCATION SUBTOTAL IS   $      890.00

0600
       20170    OUTLET                         22505        $0.69

                             LOCATION SUBTOTAL IS   $   15,528.45

0750
       20180    SWITCH PLATE                   20500        $0.20
       20120    12AMP BREAKER                  19400        $3.50
       20200    LABELS                         18175        $0.01

                             LOCATION SUBTOTAL IS   $   72,181.75
```

BEST COPY AVAILABLE

Figure JS1-7: Sample Output of Inventory Listing by Branch Location (Part 2 of 2)



```
                    B R A N C H   I N V E N T O R Y   R E P O R T

BRANCH  PART NO.     PART DESCRIPTION        AMOUNT ON HAND   PRICE  ERROR

0800
        20125     DOOR ASSY                           20300        $30.00
        20130     BIND SCREW                          50100         $0.01
        20132     DUPLEX SWITCH                        1000        $10.00
        20135     FUSEHOLDER                          47600         $0.95
        20200     LABELS                              18175         $0.01

                           LOCATION SUBTOTAL IS     $  664,902.75

0900
        20225     CONTACTORS                          18844         $0.10

                           LOCATION SUBTOTAL IS     $    1,884.40

1000
        20235     20 VOLT POWER ASSY                  19500         $0.50
        20240     FRICTION LATCH                      19100         $0.25

                           LOCATION SUBTOTAL IS     $   14,525.00

1100
        20275     TERMINAL BLOCK                      17000         $0.50

                           LOCATION SUBTOTAL IS     $    8,500.00

                           TOTAL VALUE OF PARTS IS $1,712,821.35
END OF INVENTORY REPORT PROGRAM
```

Answer to Job Sheet 1: Complete Solution to Payroll Validation Report (Part 1 of 4)

```
IDENTIFICATION DIVISION.
PROGRAM-ID.  LISTER.

ENVIRONMENT DIVISION.
CONFIGURATION SECTION.
SOURCE-COMPUTER.  IBM-370.
OBJECT-COMPUTER.  IBM-370.
SPECIAL-NAMES.
     C01 IS TOP-OF-PAGE.
INPUT-OUTPUT SECTION.
FILE-CONTROL.
     SELECT PAYROLL-FILE ASSIGN TO UR-S-SYSIN.
     SELECT PAYROLL-LIST-FILE ASSIGN TO UR-S-SYSPRINT.
DATA DIVISION.
FILE SECTION.
FD  PAYROLL-FILE
    LABEL RECORDS ARE OMITTED.
01  PAYROLL-RECORD                        PIC X(80).
FD  PAYROLL-LIST-FILE
    LABEL RECORDS ARE OMITTED.
01  PRINT-LINE                            PIC X(121).
WORKING-STORAGE SECTION.
01  PAYROLL-DATA.
    05  EMPLOYEE-NAME                     PIC X(25).
    05  EMPLOYEE-ID                       PIC X(10).
    05  DEPARTMENT                        PIC X(15).
    05  RATE-OF-PAY                       PIC 99V99.
    05  SALARY-CODE                       PIC X.
    05  HOURS-WORKED                      PIC 99.
    05  FILLER                            PIC X(23).
01  DETAIL-LINE.
    05  FILLER                            PIC X(2).
    05  D-EMPLOYEE-NAME                   PIC X(25).
    05  FILLER                            PIC X(2).
    05  D-EMPLOYEE-ID                     PIC X(10).
    05  FILLER                            PIC X(3).
    05  D-RATE-OF-PAY                     PIC $Z9.99.
    05  FILLER                            PIC X(5).
    05  D-HOURS-WORKED                    PIC Z9.
    05  FILLER                            PIC X(4).
    05  D-REGULAR                         PIC $ZZZ9.99.
    05  FILLER                            PIC X(5).
    05  D-OVERTIME                        PIC $ZZZ9.99.
    05  FILLER                            PIC X(5).
    05  D-TOTAL-PAY                       PIC $ZZZZ9.99.
    05  FILLER                            PIC X(5).
    05  ERROR-MESSAGE                     PIC X(35).
```

```
01  REPORT-TITLE.
    05  FILLER                            PIC X(50) VALUE SPACES.
    05  FILLER                            PIC X(19) VALUE
                    'PAY RATE REPORT'.
    05  FILLER                            PIC X(53) VALUE SPACES.
01  COLUMN-HEADER.
    05  FILLER                            PIC X(2) VALUE SPACES.
    05  FILLER                            PIC X(13) VALUE
                    'EMPLOYEE NAME'.
    05  FILLER                            PIC X(14) VALUE SPACES.
    05  FILLER                            PIC X(9)  VALUE
                    'ID NUMBER'.
    05  FILLER                            PIC X(3) VALUE SPACES.
    05  FILLER                            PIC X(8)  VALUE
                    'PAY RATE'.
    05  FILLER                            PIC X(2)  VALUE SPACES.
    05  FILLER                            PIC X(5)  VALUE 'HOURS'.
    05  FILLER                            PIC X(4)  VALUE SPACES.
    05  FILLER                           PIC X(7) VALUE 'REGULAR'.
    05  FILLER                            PIC X(5)  VALUE SPACES.
    05  FILLER                            PIC X(8)  VALUE
                    'OVERTIME'.
    05  FILLER                            PIC X(5)  VALUE SPACES.
    05  FILLER                            PIC X(9)  VALUE
                    'TOTAL PAY'.
    05  FILLER                            PIC X(5)  VALUE SPACES.
    05  FILLER                            PIC X(13) VALUE
                    'ERROR MESSAGE'.
01  TOTAL-LINE.
    05  FILLER                            PIC X(2) VALUE SPACES.
    05  FILLER                            PIC X(10) VALUE
                       'TOTAL FOR '.
    05  D-EMP-DEPT-A                      PIC X(14).
    05  FILLER                           PIC X(7) VALUE ' EQUALS'.
    05  FILLER                            PIC X(25) VALUE SPACES.
    05  T-REGULAR                         PIC $ZZZZ9.99.
    05  FILLER                            PIC X(4)  VALUE SPACES.
    05  T-OVERTIME                        PIC $ZZZZ9.99.
    05  FILLER                            PIC X(4)  VALUE SPACES.
    05  T-TOTAL-PAY                       PIC $ZZZZZ9.99.
01  END-TOTAL-LINE.
    05  FILLER                            PIC X(2)  VALUE SPACES.
    05  FILLER                            PIC X(26) VALUE
                       'TOTALS FOR THE COMPANY ARE'.
    05  FILLER                            PIC X(30) VALUE SPACES.
    05  E-C-PEG                           PIC $ZZZZ9.99.
    05  FILLER                            PIC X(4)  VALUE SPACES.
```

BEST COPY AVAILABLE

BEST COPY AVAILABLE

Answer to JS1: Complete Solution to Payroll Validation Report (Part 2 of 4)

```
           05  E-C-OVER                      PIC $ZZZZ9.99.
           05  FILLER                        PIC X(4)  VALUE SPACES.
           05  E-C-TOTAL                     PIC $ZZZZZ9.99.
       01  BLANK-LINE.
           05  FILLER                       PIC X(121) VALUE SPACES.
       01  END-LINE.
           05  FILLER                        PIC X(2)  VALUE SPACES.
           05  FILLER                        PIC X(14) VALUE
                                'END OF REPORT'.
       01  DEPT-LINE.
           05  FILLER                        PIC X(2)  VALUE SPACES.
           05  D-EMP-DEPT                    PIC X(15) VALUE SPACES.
           05  FILLER                        PIC X(105) VALUE SPACES.
       01  C-TOTAL                          PIC 99999V99  VALUE ZERO.
       01  C-REG                            PIC 99999V99  VALUE ZERO.
       01  C-OVER                           PIC 99999V99  VALUE ZERO.
       01  TOTAL-DEPT                        PIC 9999V99  VALUE ZERO.
       01  OVER-DEPT                        PIC 9999V99   VALUE ZERO.
       01  REG-DEPT                         PIC 9999V99   VALUE ZERO.
       01  REGULAR                          PIC 9999V99   VALUE ZERO.
       01  OVERTIME                         PIC 9999V99   VALUE ZERO.
       01  TOTAL-PAY                        PIC 9999V99   VALUE ZERO.
       01  LINE-COUNT                       PIC 99        VALUE ZERO.
       01  LINE-LIMIT                        PIC 99        VALUE 55.
       01  OVERTIME-HOUR                    PIC 99        VALUE ZERO.
       01  REGULAR-HOURS                     PIC 99       VALUE ZERO.
       01  PREVIOUS-EMP-DEPT                 PIC X(14)  VALUE SPACES.
       01  O-T-RATE                          PIC 99V99   VALUE ZERO.
       01  FORTY                             PIC 99      VALUE 40.
       01  ERROR-SWITCH                      PIC X(1)    VALUE 'N'.
           88 THERE-IS-AN-ERROR                          VALUE 'Y'.
           88 THERE-IS-NOT-AN-ERROR                      VALUE 'N'.
       01  FIRST-TIME-THRU-SWITCH            PIC X(1)    VALUE 'Y'.
           88 FIRST-TIME-THRU                            VALUE 'Y'.
           88 NOT-FIRST-TIME-THRU                        VALUE 'N'.
       01  MORE-DATA                         PIC X       VALUE 'Y'.
           88  THERE-IS-MORE-DATA                        VALUE 'Y'.
           88  THERE-IS-NO-MORE-DATA                     VALUE 'N'.
       PROCEDURE DIVISION.
       DIRECTOR-MODULE.
           PERFORM READY-FILES.
           PERFORM READ-PRINT UNTIL THERE-IS-NO-MORE-DATA.
           PERFORM CLOSE-FILES.
           STOP RUN.
       READY-FILES.
           OPEN INPUT PAYROLL-FILE
               OPEN OUTPUT PAYROLL-LIST-FILE.
           PERFORM NEW-PAGE-ROUTINE.
           MOVE SPACES TO DETAIL-LINE.
       READ-PRINT.
           READ PAYROLL-FILE INTO PAYROLL-DATA
                AT END MOVE 'N' TO MORE-DATA.
```

```
           IF THERE-IS-MORE-DATA
                PERFORM INPUT-VALIDATION
                IF PREVIOUS-EMP-DEPT NOT EQUAL DEPARTMENT
                    PERFORM EMP-DEPT-BREAK-ROUTINE.
           IF THERE-IS-MORE-DATA
                PERFORM REGULAR-ROUTINE
                PERFORM OVERTIME-ROUTINE
                COMPUTE TOTAL-PAY = OVERTIME + REGULAR.
           IF THERE-IS-MORE-DATA
                IF THERE-IS-NOT-AN-ERROR
                    COMPUTE TOTAL-DEPT = TOTAL-DEPT + TOTAL-PAY
                    COMPUTE OVER-DEPT = OVER-DEPT + OVERTIME
                    COMPUTE REG-DEPT = REG-DEPT + REGULAR
                    COMPUTE C-REG = C-REG + REGULAR
                    COMPUTE C-OVER = C-OVER + OVERTIME
                    COMPUTE C-TOTAL = C-TOTAL + TOTAL-PAY.
           IF THERE-IS-MORE-DATA
                MOVE EMPLOYEE-NAME TO D-EMPLOYEE-NAME
                MOVE EMPLOYEE-ID TO D-EMPLOYEE-ID
                MOVE HOURS-WORKED TO D-HOURS-WORKED
                MOVE RATE-OF-PAY TO D-RATE-OF-PAY
                MOVE REGULAR TO D-REGULAR
                MOVE OVERTIME TO D-OVERTIME
                MOVE TOTAL-PAY TO D-TOTAL-PAY.
           IF THERE-IS-MORE-DATA
                IF LINE-COUNT GREATER THAN LINE-LIMIT
                   PERFORM NEW-PAGE-ROUTINE.
           IF THERE-IS-MORE-DATA
                WRITE PRINT-LINE FROM DETAIL-LINE
                   AFTER ADVANCING 1 LINES
                ADD 1 TO LINE-COUNT.
       REGULAR-ROUTINE.
           IF HOURS-WORKED GREATER THAN 40
                COMPUTE REGULAR = RATE-OF-PAY * FORTY
           ELSE
               COMPUTE REGULAR = HOURS-WORKED * RATE-OF-PAY.
       OVERTIME-ROUTINE.
           MOVE ZERO TO OVERTIME.
           IF HOURS-WORKED GREATER THAN 40
               COMPUTE OVERTIME-HOUR = HOURS-WORKED - FORTY
               COMPUTE O-T-RATE = RATE-OF-PAY * 1.5
               COMPUTE OVERTIME = OVERTIME-HOUR * O-T-RATE.
       INPUT-VALIDATION.
           MOVE SPACE TO ERROR-MESSAGE.
           MOVE 'N' TO ERROR-SWITCH.
           IF EMPLOYEE-NAME NUMERIC
               MOVE 'Y' TO ERROR-SWITCH
               MOVE 'ILLEGAL NAME' TO ERROR-MESSAGE
           ELSE
               IF EMPLOYEE-ID EQUAL SPACES
                   MOVE 'Y' TO ERROR-SWITCH
                   MOVE 'ILLEGAL NUMBER' TO ERROR-MESSAGE
```

Answer to JS1: Complete Solution to Payroll Validation Report (Part 3 of 4)

```
    ELSE
      IF DEPARTMENT NOT ALPHABETIC
        MOVE 'Y' TO ERROR-SWITCH
        MOVE 'INVALID DEPT' TO ERROR-MESSAGE
      ELSE
        IF RATE-OF-PAY NOT NUMERIC
          MOVE 'Y' TO ERROR-SWITCH
          MOVE 'INVALID RATE' TO ERROR-MESSAGE
        ELSE
          IF HOURS-WORKED NOT NUMERIC
            MOVE 'Y' TO ERROR-SWITCH
            MOVE 'INVALID HOURS' TO ERROR-MESSAGE.
EMP-DEPT-BREAK-ROUTINE.
  IF FIRST-TIME-THRU
    MOVE 'N' TO FIRST-TIME-THRU-SWITCH
    PERFORM NEW-EMP-DEPT-ROUTINE
  ELSE
    PERFORM OLD-EMP-DEPT-ROUTINE
    PERFORM NEW-EMP-DEPT-ROUTINE.
OLD-EMP-DEPT-ROUTINE.
  MOVE PREVIOUS-EMP-DEPT TO D-EMP-DEPT-A.
  MOVE DEPARTMENT TO PREVIOUS-EMP-DEPT
  MOVE REG-DEPT TO T-REGULAR
  MOVE OVER-DEPT TO T-OVERTIME
  MOVE TOTAL-DEPT TO T-TOTAL-PAY
  IF LINE-COUNT GREATER THAN LINE-LIMIT
    PERFORM NEW-PAGE-ROUTINE.
  WRITE PRINT-LINE FROM TOTAL-LINE
     AFTER ADVANCING 2 LINES.
  ADD 6 TO LINE-COUNT.
  MOVE ZERO TO REGULAR.
  MOVE ZERO TO OVERTIME.
```

```
  MOVE ZERO TO TOTAL-PAY.
NEW-EMP-DEPT-ROUTINE.
  MOVE DEPARTMENT TO D-EMP-DEPT.
  MOVE DEPARTMENT TO PREVIOUS-EMP-DEPT.
  IF LINE-COUNT GREATER THAN LINE-LIMIT
    PERFORM NEW-PAGE-ROUTINE.
  WRITE PRINT-LINE FROM DEPT-LINE
    AFTER ADVANCING 2 LINES.
  WRITE PRINT-LINE FROM BLANK-LINE
    AFTER ADVANCING 1 LINES.
  MOVE 6 TO LINE-COUNT.
  MOVE ZERO TO TOTAL-DEPT.
  MOVE ZERO TO REG-DEPT.
  MOVE ZERO TO OVER-DEPT.
NEW-PAGE-ROUTINE.
  WRITE PRINT-LINE FROM REPORT-TITLE
    AFTER ADVANCING TOP-OF-PAGE.
  WRITE PRINT-LINE FROM COLUMN-HEADER
    AFTER ADVANCING 2 LINES.
  WRITE PRINT-LINE FROM BLANK-LINE
    AFTER ADVANCING 1 LINES.
  MOVE 4 TO LINE-COUNT.
CLOSE-FILES.
  PERFORM OLD-EMP-DEPT-ROUTINE.
  MOVE C-REG TO E-C-REG.
  MOVE C-OVER TO E-C-OVER.
  MOVE C-TOTAL TO E-C-TOTAL.
  WRITE PRINT-LINE FROM END-TOTAL-LINE
    AFTER ADVANCING 4 LINES.
  WRITE PRINT-LINE FROM END-LINE
    AFTER ADVANCING 3 LINES.
  WRITE PRINT-LINE FROM BLANK-LINE
    AFTER ADVANCING 4 LINES.
  CLOSE PAYROLL-FILE   PAYROLL-LIST-FILE.
```

BEST COPY AVAILABLE

Answer to JS1: Complete Solution to Payroll Validation Report (Part 4 of 4)

```
                                          PAY RATE REPORT
EMPLOYEE NAME            ID NUMBER   PAY RATE  HOURS   REGULAR     OVERTIME     TOTAL PAY    ERROR MESSAGE

LIBRARY

ROGER ADAMS              643-128-32  $ 4.25     50    $ 170.00    $  63.70    $  233.70
WILLIE NELSON            444-444-44  $ 3.35     40    $ 134.00    $   0.00    $  134.00

TOTAL FOR LIBRARY        EQUALS                     $  304.00   $   63.70   $   367.70

MAINTENANCE

RON JENKINS              704-403-98  $ 5.50     35    $ 192.50    $   0.00    $  192.50
DALE JANSON, JR          621-734-82  $ 6.75     45    $ 270.00    $  50.60    $  320.60

TOTAL FOR MAINTENANCE    EQUALS                     $  462.50   $   50.60   $   513.10

MARKETING

TARA EDDY                333-333-33  $ 7.00     30    $ 210.00    $   0.00    $  210.00
JOHN SCHROEDER           618-395-24  $ 3.50     40    $ 140.00    $   0.00    $  140.00

TOTAL FOR MARKETING      EQUALS                     $  350.00   $    0.00   $   350.00

ADVERTISING

LAURA JAMES              308-215-43  $ 6.50     45    $ 260.00    $  48.75    $  308.75
TEST                                 $ 5.00     40    $ 200.00    $   0.00    $  200.00    ILLEGAL NUMBER

TOTAL FOR ADVERTISING    EQUALS                     $  260.00   $   48.75   $   308.75

SYSTEMS DEVPMT

ARTHUR WEBER             712-561-34  $ 5.50     30    $ 165.00    $   0.00    $  165.00
DON WILLIAMS             111-111-11  $ 3.35     40    $ 134.00    $   0.00    $  134.00

TOTAL FOR SYSTEMS DEVPMT EQUALS                     $  299.00   $    0.00   $   299.00

PRODUCTION

JOHN CHANCELLOR          645-552-21  $ 6.50     55    $ 260.00    $ 146.25    $  406.25
SUE TOLEN                222-222-22  $99.99     50    $3999.60    $ 499.80    $ 4499.40

TOTAL FOR PRODUCTION     EQUALS                     $ 4259.60   $  646.05   $  4905.65

PURCHASING

DEBRORAH JOHNAS          832-481-86  $ 4.95     40    $ 198.00    $   0.00    $  198.00

TOTAL FOR PURCHASING     EQUALS                     $  198.00   $    0.00   $   198.00

ACCOUNTING

JAMES E. LANG            712-825-76  $ 3.75     40    $ 150.00    $   0.00    $  150.00
JUDITH S. PARKS          542-861-79  $ 7.50     25    $ 187.50    $   0.00    $  187.50
TEST                     999-999-99  $ 6.00     13    $  78.00    $   0.00    $   78.00    INVALID HOURS

TOTAL FOR ACCOUNTING     EQUALS                     $  337.50   $    0.00   $   337.50



MUSIC

BOOTS RANDOLPH           555-555-55  $ 3.35     50    $ 134.00    $  50.20    $  184.20

TOTAL FOR MUSIC          EQUALS                     $  134.00   $   50.20   $   184.20


TOTALS FOR THE COMPANY ARE                          $ 6604.60   $  859.30   $  7463.90

END OF REPORT
```

CONTROL BREAKS

UNIT VI

EXAMINATION

1. Match each phrase in Column B with its correct term in Column A:

		Column A		Column B
____	a.	Department Accumulator	1.	Sorted on this field
____	b.	Control break	2.	Unequal record keys
____	c.	COMP-3	3.	Used for Department Total
____	d.	Key field	4.	Packed decimal format

2. Was there a control break in the sample program on the first card?

3. Which module calculates inventory value?

4. Is the second record a control break in the sample program?

5. Why is the Department Accumulator set to zero when a Department Control Break occurs?

6. Why is the Department Total printed when the card reader is empty?

7. When there is a change in Department Number, why is Department Total to be printed?

8. What is the name of the submodule coded to the CONTROL INVENTORY REPORT MODULE? Why wasn't it drawn in the IPO Charts?

9. The following statements are either true or false. Circle "T" if the statement is True; circle "F" if the statement is False.

T F a. The key field in the sample program consists of Department Number.

T F b. The Department Number of the first control break in the sample program is 101.

T F c. The Department Number does not have to be inputted to the control break program on every card.

T F d. Part number is printed only the first time it appears in the report.

T F e. The module named "Process Detail Records" is a second-level module in the sample program.

T F f. Three second-level modules are planned for in the sample program.

T F g. The Control Inventory Report module opens and closes the files.

T F h. If Department Number of the inputted card is not equal to the Department Number in the compare area, the sample program's logic calls for a department change to be processed.

CONTROL BREAKS

UNIT VI

ANSWERS TO EXAMINATION

1. a. 3
   b. 2
   c. 4
   d. 1

2. No

3. Process Detail Records Module

4. No

5. The Department Accumulator is set to zero when a Department Control Break occurs to initialize its value for the following department; i.e., so that the first department's totals do not get mixed into those of the second department.

6. The card reader empty means that the end of job has occurred. This occurs during the processing of the last department; thus it is necessary to finish the last department by printing its totals.

7. As this is the control break, the control break line on department is to be printed.

8. A001-COMPARE-AND-PROCESS. It wasn't drawn in the documentation as it is short and simple and was written to make its driver module simpler.

9. a. T
   b. F
   c. F
   d. F
   e. T
   f. T
   g. T
   h. T

MULTIPLE-LEVEL CONTROL BREAKS

UNIT VII

UNIT OBJECTIVES

After studying this unit, the student should be able to utilize multiple-level control break processing techniques.

SPECIFIC OBJECTIVES

After studying this unit, the student should be able to:

1. Match terms and definitions.
2. Explain concepts underlying the design of multiple-level control break programs.
3. Identify control keys and sequence for a sample program utilizing multiple-level control breaks.
4. Identify fields used in initial processing.
5. List steps in processing of detail records.
6. Explain steps in processing of control breaks for four levels.
7. Explain details of sample program design in the Control Sales Report Module.
8. Identify functions as in the IPO Chart processing steps of the Process Detail Records Module, Process Customer Change Module, Process Salesman Change Module, Process Branch Change Module, and Print Final Total Module.
9. Identify programming details in the pseudocode specifications for the Control Sales Report Module, Process Detail Records Module, Process Customer Change Module, Process Salesman Change Module, and Process Branch Change Module.
10. Identify programming techniques in the coding of a sample program.

MULTIPLE-LEVEL CONTROL BREAKS

UNIT VII

SUGGESTED ACTIVITIES

I. Instructor
    A. Make transparencies.
    B. Provide students with objectives and discuss.
    C. Provide students with information and assignment sheets; discuss these sheets.
    D. Make parallel assignments for reading and programming from text.
    E. Provide supplementary information and assignments.
    F. Give examination.

II. Student
    A. Read objectives.
    B. Study information sheets and text assignments.
    C. Complete assignment sheets and supplementary assignments.
    D. Complete examination.

INSTRUCTIONAL MATERIALS

I. Unit content
    A. Objectives
    B. Information sheets
    C. Handouts
        1. HO 1: Programming Specifications
        2. HO 2: Pseudocode for Control Sales Report Module
        3. HO 3: Pseudocode Specifications for Process Detail Records Module
        4. HO 4: Pseudocode Specifications for Process Customer Change Module
        5. HO 5: Pseudocode Specifications for Process Salesman Change Module

    6. HO 6: Pseudocode Specifications for Process Branch Change Module

D. Transparency masters

    1. TM 1: Multiple-Level Control Break Report with Group Indication
    2. TM 2: Printer Spacing Chart and Input Record Format
    3. TM 3: Processing the First Card as a Detail Record
    4. TM 4: IPO Chart and Hierarchy Chart for Sample Program
    5. TM 5: IPO Chart - Process Detail Records Module
    6. TM 6: IPO Chart - Process Customer Change Module
    7. TM 7: IPO Chart - Process Salesman Change Module
    8. TM 8: IPO Chart - Process Branch Change Module
    9. TM 9: IPO Chart - Print the Final Total Module
    10. TM 10: Group-Indicated Report Sample
    11. TM 11: COBOL Coding - Control Sales Report Module
    12. TM 12: COBOL Coding - Process Detail Record Module
    13. TM 13: Source Listing (Part 1 and Part 2)

E. Assignment sheets

    1. AS 1: Terms and Introduction to Multiple-Level Control Breaks
    2. AS 2: Overview of Sample Program, Initial, and Detailed Processing
    3. AS 3: Actual Processing of the Four Levels of Control Breaks
    4. AS 4: IPO Charts for Sample Program Modules
    5. AS 5: Pseudocode for Sample Program Modules

F. Answers to assignment sheets

G. Examination

H. Answers to examination

II. Reference

A. Shelly, Gary B. and Thomas J. Cashman. "Introduction to Computer Programming." in Structured COBOL. Fullerton, CA: Anaheim Publishing Company, 1977.

MULTIPLE-LEVEL CONTROL BREAKS

UNIT VII

INFORMATION SHEETS

I. Terms and functional definitions

- A. **Multiple-level control break** - control breaks which are hierarchically contained within one another
- B. **Group-indicated report items** - repetitive items on reports which are only printed the first time they occur--generally control break information
- C. **Control key (record key, sort key)** - nearly synonymous with control break; it is the record key or sort key which then becomes the break field in a report or file
- D. **Driver module** - the top level module in a hierarchy chart--controls or drives the lower level modules

II. Introduction to multiple-level control breaks

- A. Example of multiple-level control breaks:

  Customer number within

  Salesman within

  Branch within

  Total report

- B. Group indicated means that only the first occurrence of repetitive information is printed. (TM 1)
- C. Sorting - multiple-level breaks require multiple-level sorts of input file. In this example:
  - 1. Most major field = branch number
  - 2. Second most major field = salesman number
  - 3. Most minor field = customer number
- D. When a higher level control break occurs, the lower levels are automatically assumed to break; e.g., if a break on branch number occurs, any Customer Total or Salesman Total would naturally be assumed to break also.

III. Sample program using multiple-level control breaks - overview

- A. Input and output specifications (TM 2)
- B. Programming specifications (HO 1)

IV. Multiple-level control break processing - initial processing

- A. Initial processing (TM 3)
  - 1. Similar to example from preceding unit, but
  - 2. Since Branch Number, Customer Number, and Salesman Number are all control keys, each must be moved to the Control Area to set up the contents of the Control Area for Control Key Comparison.

V. Multiple-level control break processing - processing of detailed records

- A. Similar to example from preceding unit; steps are:
  - 1. As long as the record keys of the record read are equal to the control keys of the Compare Area, the processing is simply a detail record. The main steps are:
    - a. Set up output area
    - b. Accumulate sales total to:
      - 1) Branch Total Accumulator
      - 2) Salesman Total Accumulator
      - 3) Customer Total Accumulator
      - 4) Final Total Accumulator
    - c. Print the detail record to finish process for a record which is not a control break.

VI. Multiple-level control break processing - processing of control breaks (record keys unequal to compare area)

- A. Lowest-level control break
  - 1. In the example, the most minor key is customer. When it changes, this is a control break for a customer.
  - 2. Program actions
    - a. Move Customer Total Amount to Customer Total Line and print.
    - b. Set Customer Total to zero.

c. Move new Customer Number to Customer Number in the Compare Area.

d. Process record which caused the break as a detail record.

B. Second-level control breaks

1. In this example the second-level of control is Salesman. When Salesman changes, this is a control break on Salesman.

2. Program actions at this break

    a. Move Customer Total amount to Customer Total Line and print it; i.e., the higher-level break automatically causes the lower-level break.

    b. Move the Salesman Total to Salesman Total Line and print. Zero Salesman Total and move Salesman Number and Customer Number from the input area to the Compare Area.

    c. Process the record in the input area as a detail record.

C. Third-level control breaks

1. In this example, the third level of control is Branch Number. When Branch Number changes, there is a third-level control break, so there are automatic breaks at the second and first levels.

2. Program actions

    a. Do first-and-second level control breaks with printing of the two lines.

    b. Set up Branch Total Line and print it. Zero Branch Total. Set up Customer, Salesman, and Branch Numbers in the Compare Areas.

    c. Process the Input Record as a Detail Record.

D. Final total/end of job control breaks

1. Occurs after all records have been processed but their accumulated total must be processed. This causes all three lower-level breaks to occur also.

2. Program actions

    a. Do first-, second-, and third-level control breaks with printing of the three lines.

    b. Move Final Total to Final Total Line and print it.

    c. Terminate program.

VII. Program Design - IPO chart - Control Sales Report Module (TM 4)

A. Input is Sales Input File. Output is Sales Report.

B. Specific processing steps are:

1. Initialization
2. Obtain the input data.
3. Process detail records.
4. Process customer changes.
5. Process salesman changes.
6. Process branch changes.
7. Print the final totals
8. Termination

C. Note that each major processing step is a second-level module in TM 4, the hierarchy chart.

D. The next step in design is to break down each of the second-level modules by means of an IPO Chart for each module.

VIII. Program design - IPO Chart - detail module

A. Program design - IPO Chart - Process Detail Records Module (TM 5)

1. Module's input is Sales File and Accumulators. Outputs are the printed lines and updated accumulators.

2. Major processing steps:

   a. Print report header.
   b. Format detail lines.
   c. Update accumulators.
   d. Print detail line.

3. Upon inspection of the complexity of the secondary module, it was determined that no further submodule except a semi-standard print headings module is required.

B. Program Design - IPO Chart - Process Customer Change Module (TM 6)

1. This is the first-level break module. Input is new Customer Number and Customer Total Accumulator. Output is a Customer Total Line and Customer Total Accumulator.

2. Major processing steps:

   a. Format Customer Total Line and print it.

   b. Zero Customer Total Accumulator.

   c. Move Customer Number into compare area to set up for next control break.

C. Program design - IPO Chart - Process Salesman Change Module (TM 7)

1. This is a module to accomplish the second-level break on Salesman Number. Input is the new Salesman Number and Salesman Total Accumulator. Outputs are the Salesman Total Line, Total Accumulator, and updated Compare Area.

2. Major processing steps:

   a. Format Salesman Total Line and print it.

   b. Zero Salesman Total Accumulator and move new Salesman Number to Compare Area.

D. Program Design - IPO Chart - Process Branch Change Module (TM 8)

1. This is the module for the third-level control break on the Branch Number Change. Input is the changed Branch Number and Branch Total Accumulator. Outputs are the Branch Total Line, Total Accumulator, and updated Compare Area.

2. Major processing steps:

   a. Format Branch Total Line and print it.

   b. Zero Branch Total Accumulator and move New Branch Number into Compare Area.

E. Program design - IPO Chart - Print Final Total Module (TM 9)

1. This is the module for the overall level control break at the end of job. Input is the Final Total Accumulator only. Output is the Final Total Line.

2. Major processing steps are format the Final Total Line, print it, and then terminate processing.

IX. Pseudocode specifications

A. Pseudocode specifications - Control Sales Report Module (HO 2)

1. This is a complex program, which requires systematic study; but the IPO Charts and the pseudocode specifications break down the complexities into independent modules, which can be easily understood in one sitting. So, study each module with IPO Chart and pseudocode in hand.

2. Pseudocode specifications major notes

   a. This is the highest-level module in the hierarchy chart and controls all the other modules.

   b. The module uses the READ-BEFORE-PROCESSING technique.

   c. Notice the control break hierarchy embedded in the nested IF/ELSE sentence. In pseudocode this can be rewritten as:

```
IF highest (third) level break
    DO 1st level module
    DO 2nd level module
    DO 3rd level module
ELSE
    IF 2nd level break
    DO 1st level module
    DO 2nd level module
ELSE
    IF 1st level break
    DO  1st level module
```

B. Pseudocode specifications - Process Detail Records Module (HO 3)

1. This module does not necessarily involve a control break, but always processes a detail record.

2. Notes:

   a. This module controls the **group indicated** print feature with a complex IF/ELSE statement. That statement could be rewritten as:

```
IF Branch break
    move Branch, Salesman and Customer to Print
ELSE
    IF Salesman break move Salesman and Customer to Print
    ELSE
        IF Customer break
            Move Customer to Print.
```

b. Because the print area is always cleared to spaces after each print, the above statement results in the selective printing of the group indicated control keys. (TM 10)

C. Pseudocode specifications - Process Customer Change Module (HO 4)

1. This is the module which processes the lowest-level control break, the customer number.
2. This and other modules do tneir own printing and line control. This is often done by other designers in a separate print-control module.

D. Pseudocode specifications - Process Salesman Change Module (HO 5)

1. Similar to B010 module but on second-level control break, the salesman number

E. Pseudocode specifications - Process Branch Change Module (HO 6)

1. Same as B010 and B020 but on third-level
2. Final total module same as preceding breaks but not necessary to reset accumulators
3. Print headings module similar to previous programs

X. Sample program coding

A. Coding the Control Sales Report Module (TM 11)

1. This is the main module, which is often called the driver module.
2. Notice that the A001-PROCESS-AND READ Module has simplified the complex IF/ELSE statement by moving it to a separate paragraph. In this paragraph, this is a matched, nested IF. If Branch Number test is true, the next test will not be tested. If the test is true, do the condition and then go to the next sentence.

B. Coding the Process Detail Records Module (TM 12) - notice the complex nested IF beginning on line 013190. This is the nested IF which controls the group indicated feature.

C. Source listing of example program (TM 13)

239

HO 1

Programming Specifications

PROGRAMMING SPECIFICATIONS			
SUBJECT Sales Report		**DATE** June 16	**PAGE 1 OF 1**
TO Programmer		**FROM** Systems Analysts	

A program is to be written to prepare a Sales Report. The format of the input card file and the printer spacing chart are given on TM 2. The program should include the following processing.

1. The program should read the Sales Cards and create the Sales Report as per the format illustrated on the printer spacing chart.
2. Headings should be printed on the first and subsequent pages of the report. The heading is to contain the current date and the page number. Fifty-five detail and total lines are to appear on each page before skipping to a new page.
3. One line is to be printed on the report for each card which is read. The lines are to be single spaced.
4. When there is a change in Customer Number, a Customer Total is to be printed. When there is a change in Salesman Number, a Customer Total and Salesman Total are to be printed; and when there is a change in Branch Number, Customer, Salesman, and Branch Totals are to be printed. A Final Total is to be printed after all cards have been processed.
5. The program is to be written in COBOL.
6. Input records must be sorted by Customer Number within Salesman Number within Branch Number prior to processing.

HO 2

Pseudocode for Control Sales Report Module

Pseudocode Specifications

DATE: June 17	PROGRAMMER: Jane Austin	PROGRAM: Sales Report
MODULE FUNCTION: Control the Sales Report	REF: A000	MODULE NAME: Control Sales Report

PSEUDOCODE / **REF:**

```
Open the files
Read an input record
IF there is a record
    Move customer number to compare area
    Move salesman number to compare area
    Move branch number to compare area
    PERFORM UNTIL there are no more records
        IF branch number not = branch number in
                compare area
            Process customer change                     B010
            Process salesman change                     B020
            Process branch change                       B030
        ELSE
            IF salesman number not = salesman number in
                    compare area
                Process customer change                 B010
                Process salesman change                 B020
            ELSE
                IF customer number not = customer number
                        in compare area
                    Process customer change             B010
                ENDIF
            ENDIF
        ENDIF
        Process detail record                           B000
        Read an input record
    ENDPERFORM
    Process customer change                             B010
    Process salesman change                             B020
    Process branch change                               B030
    Print final total                                   B040
ENDIF
Close Files
Stop run
```

FILES, RECORDS, FIELDS REQUIRED

Sales Input File
Sales Report File
Input area
    Branch number
    Salesman number
    Customer number
    Description
    Sales amount
More records indicator
Customer number compare area
Salesman number compare area
Branch number compare area
No more records indicator

Pseudocode Specifications for Process Detail Records Module

Pseudocode Specifications

DATE: June 17	PROGRAMMER: Jane Austin	PROGRAM: Sales Report
MODULE FUNCTION: Process Detail Record	REF: B000	MODULE NAME: Process Detail Record

PSEUDOCODE

```
If lines printed is = or   page size or this is the
                first page
   Print the headings
   Move branch number to printer output area
   Move salesman number to printer output area
   Move customer number to printer output area
ENDIF
If there was a change in branch number
   Move branch number to printer output area
   Move salesman number to printer output area
   Move customer number to printer output area
ELSE
   If there was a change in salesman number
      Move salesman number to printer output area
      Move customer number to printer output area
   ELSE
      If there was a change in customer number
         Move customer number to printer output area
      ENDIF
   ENDIF
ENDIF
Move description to printer output area
Move sales to printer output area
Add sales to customer total accumulator, salesman total
                accumulator, branch total accumulator, final
                total accumulator
Write a line in the report
Increment lines printed
Set spacing for single spacing
Clear printer output area
```

REF: C000

FILES, RECORDS, FIELDS REQUIRED

- Lines printed counter
- Page size constant
- First page indicator
- Printer output area
  - Branch number
  - Salesman number
  - Customer number
  - Description
  - Sales
- Branch change indicator
- Current branch number
- Current salesman number
- Current customer number
- Salesman change indicator
- Customer change indicator
- Description
- Sales
- Customer total accumulator
- Salesman total accumulator
- Branch total accumulator
- Final total accumulator
- Space control field
- Single space control character

HO 4

Pseudocode Specifications for Process Customer Change Module

Pseudocode Specifications

DATE: June 17
PROGRAMMER: Jane Austin
PROGRAM: Sales Report
MODULE FUNCTION: Process Customer Change
REF: B010
MODULE NAME: Process Customer Change

PSEUDOCODE

```
Move customer total to customer total line
Write customer total line on the report
Set customer total accumulator to zero
Move the new customer number to the compare area
Add 2 to the number of lines printed
Set spacing for double spacing
```

REF:

FILES, RECORDS, FIELDS REQUIRED

- Customer total accumulator
- Customer total line
- New customer number from input
- Customer number compare area
- Number of lines printed counter
- Space control field
- Double space control character

HO 5

Pseudocode Specifications for Process Salesman Change Module

Pseudocode Specifications

DATE: June 17	**PROGRAMMER:** Jane Austin	**PROGRAM:** Sales Report
MODULE FUNCTION: Process Salesman Change	**REF:** B020	**MODULE NAME:** Process Salesman Change

PSEUDOCODE

```
Move previous salesman number to salesman total line
Move salesman total to salesman total line
Write salesman total line on the report
Set salesman total accumulator to zero
Move the new salesman number to the compare area
Add 2 to the number of lines printed
Set spacing for triple spacing
```

REF:

FILES, RECORDS, FIELDS REQUIRED

Previous salesman number from compare area
Salesman total accumulator
Salesman total line
New salesman number from input record
Salesman number compare area
Number of lines printed counter
Space control field
Triple space control character

HO 6

Pseudocode Specifications for Process Branch Change Module

Pseudocode Specifications

DATE: June 17	PROGRAMMER: Jane Austin	PROGRAM: Sales Report
MODULE FUNCTION: Process Branch Change	REF: B030	MODULE NAME: Process Branch Change

PSEUDOCODE

```
Move previous branch number to branch total line
Move branch total to branch total line
Write branch total line on the report
Set branch total accumulator to zero
Move the new branch number to the compare area
Add 2 to lines printed
Set spacing for triple spacing
```

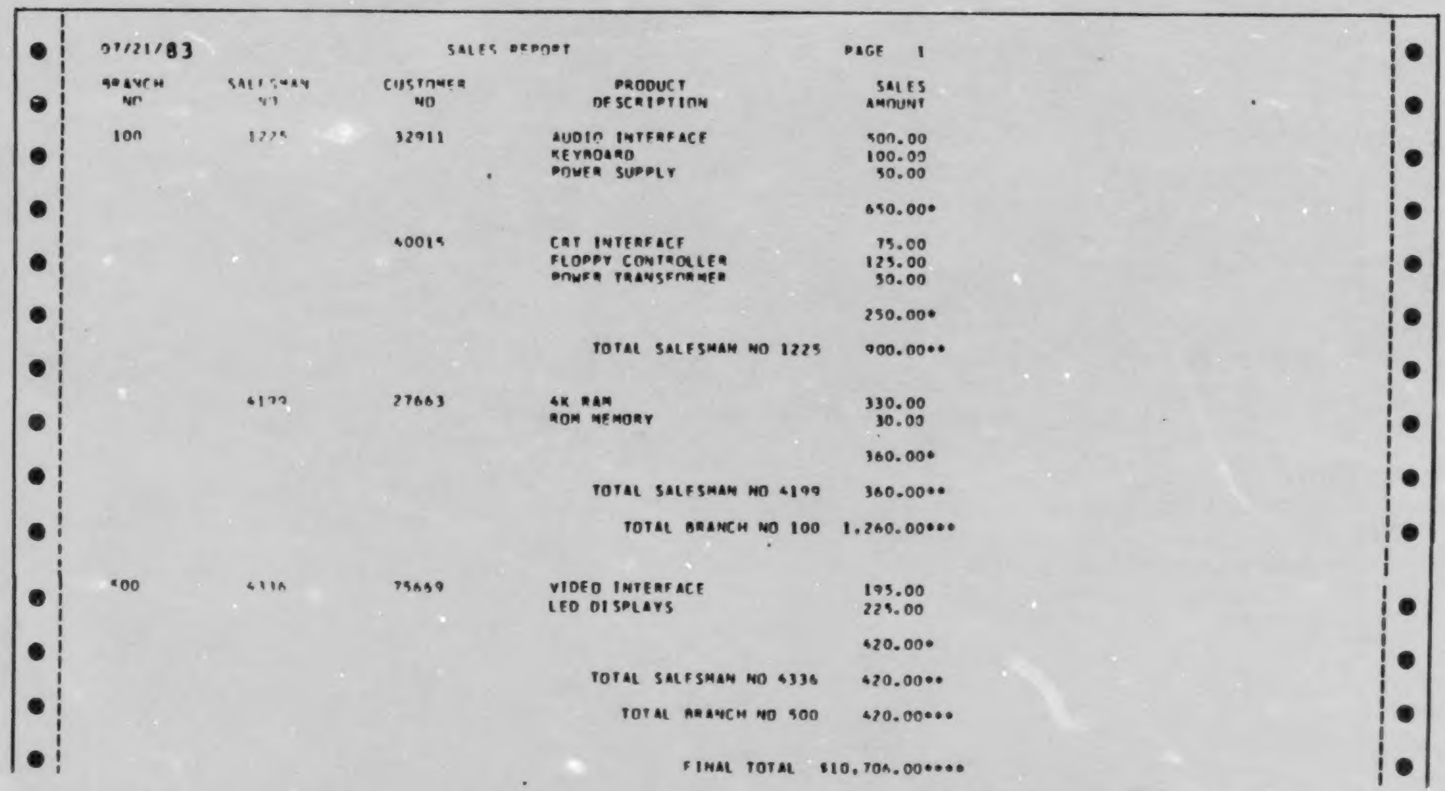
REF:

FILES, RECORDS, FIELDS REQUIRED

Previous branch number from compare area
Branch total accumulator
Branch total line
New branch number from input record
Branch number compare area
Number of lines printed counter
Space control field
Triple space control character

BEST COPY AVAILABLE

Multiple-Level Control Break Report with Group Indication

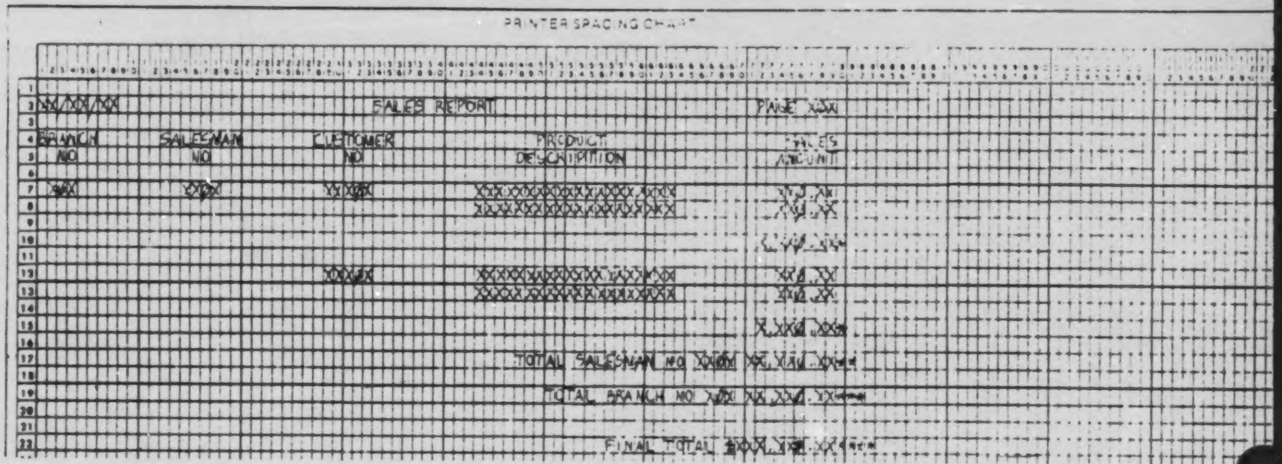


```
07/21/83                          SALES REPORT                        PAGE   1

BRANCH      SALESMAN      CUSTOMER             PRODUCT                SALES
  NO           NO            NO              DESCRIPTION              AMOUNT

 100          1225          32911         AUDIO INTERFACE             500.00
                                          KEYBOARD                    100.00
                                          POWER SUPPLY                 50.00

                                                                      650.00*

                            40015         CRT INTERFACE                75.00
                                          FLOPPY CONTROLLER           125.00
                                          POWER TRANSFORMER            50.00

                                                                      250.00*

                                             TOTAL SALESMAN NO 1225   900.00**

              4199          27663         4K RAM                      330.00
                                          ROM MEMORY                   30.00

                                                                      360.00*

                                             TOTAL SALESMAN NO 4199   360.00**

                                                TOTAL BRANCH NO 100 1,260.00***

 500          4336          75669         VIDEO INTERFACE             195.00
                                          LED DISPLAYS                225.00

                                                                      420.00*

                                             TOTAL SALESMAN NO 4336   420.00**

                                                TOTAL BRANCH NO 500   420.00***

                                                     FINAL TOTAL $10,706.00****
```

BEST COPY AVAILABLE

Printer Spacing Chart and Input Record Format

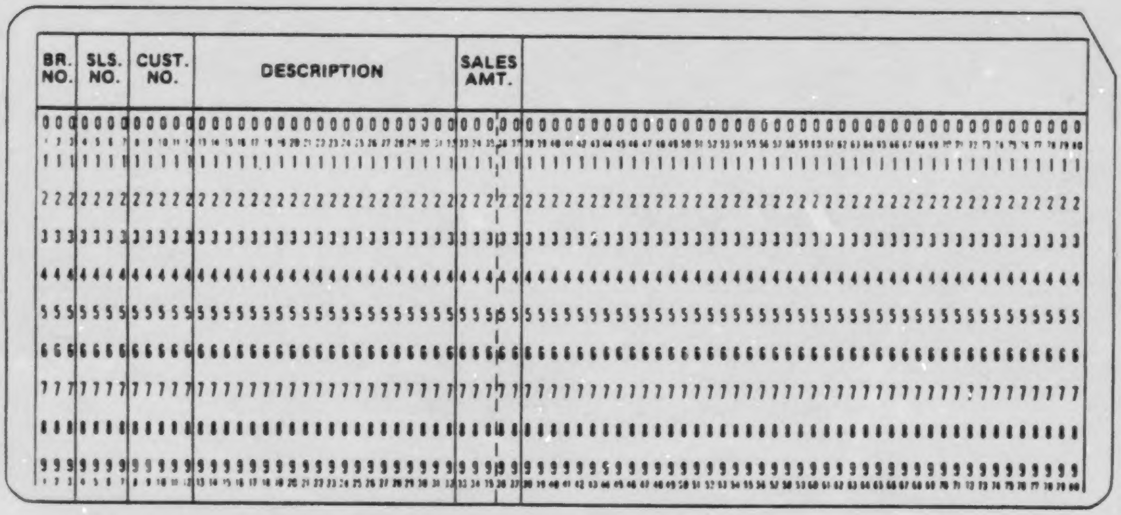
OUTPUT



PRINTER SPACING CHART

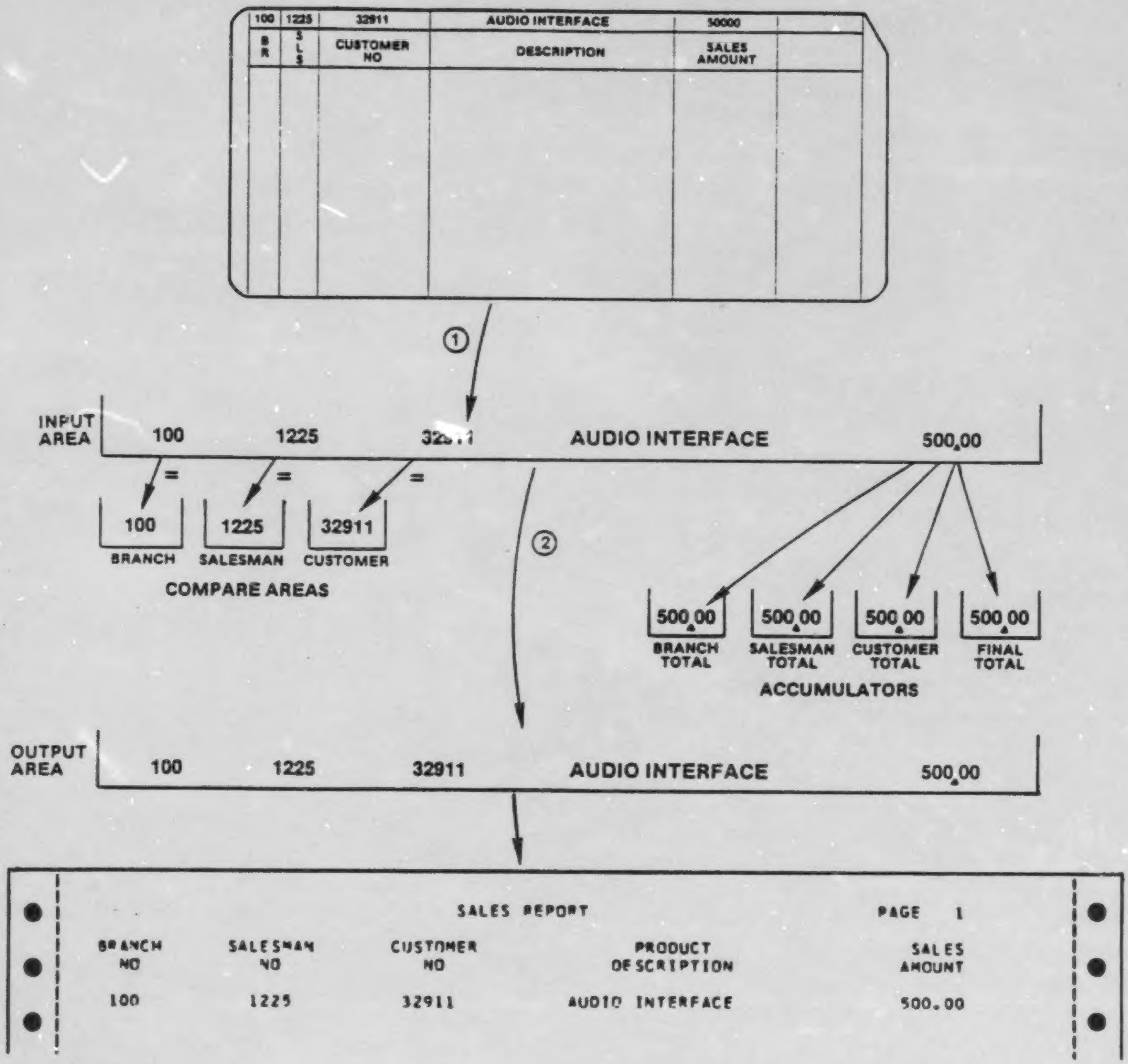
```
 1
 2 XX/XX/XX                              SALES REPORT                   PAGE XXX
 3
 4 BRANCH      SALESMAN      CUSTOMER            PRODUCT                  SALES
 5   NO          NO            NO              DESCRIPTION               AMOUNT
 6
 7  XXX          XXX          XXXX        XXXXXXXXXXXXXXXXXXXX            XXX.XX
 8                                        XXXXXXXXXXXXXXXXXXXX            XXX.XX
 9
10                                                                      X,XXX.XX*
11
12                            XXXX        XXXXXXXXXXXXXXXXXXXX            XXX.XX
13                                        XXXXXXXXXXXXXXXXXXXX            XXX.XX
14
15                                                                      X,XXX.XX*
16
17                                     TOTAL SALESMAN NO XXX  XX,XXX.XX**
18
19                                        TOTAL BRANCH NO XXX  XX,XXX.XX***
20
21
22                                              FINAL TOTAL $XXX,XXX.XX****
```

INPUT



BR. NO.	SLS. NO.	CUST. NO.	DESCRIPTION	SALES AMT.	
1–3	4–7	8–12	13–32	33–37	38–80

Processing the First Card as a Detail Record


100
1225
32911
AUDIO INTERFACE
50000
BR
SLS
CUSTOMER NO
DESCRIPTION
SALES AMOUNT
①
INPUT AREA
100
1225
32911
AUDIO INTERFACE
500.00
=
=
=
100
1225
32911
BRANCH
SALESMAN
CUSTOMER
COMPARE AREAS
②
500.00
500.00
500.00
500.00
BRANCH TOTAL
SALESMAN TOTAL
CUSTOMER TOTAL
FINAL TOTAL
ACCUMULATORS
OUTPUT AREA
100
1225
32911
AUDIO INTERFACE
500.00
SALES REPORT
PAGE 1
BRANCH NO
SALESMAN NO
CUSTOMER NO
PRODUCT DESCRIPTION
SALES AMOUNT
100
1225
32911
AUDIO INTERFACE
500.00

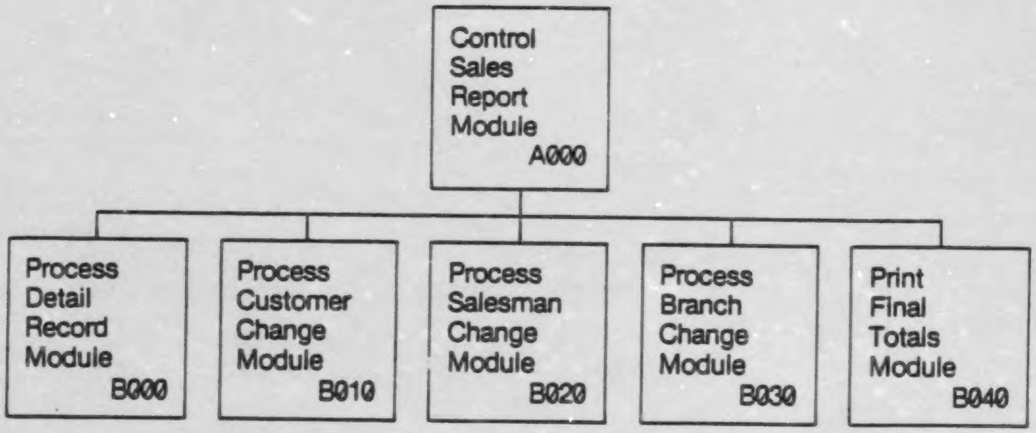
TM 4

IPO Chart and Hierarchy Chart for Sample Program

Input/Processing/Output Chart

DATE: June 17
PROGRAMMER: Jane Austin
PROGRAM: Sales Report
MODULE FUNCTION: Create the Sales Report
REF: A000
MODULE NAME: Control Sales Report

INPUT	PROCESSING	OUTPUT
1. Sales Input File	1. Initialization	1. Sales Report
	2. Obtain the input data	
	3. Process detail records	
	4. Process customer changes	
	5. Process salesman changes	
	6. Process branch changes	
	7. Print the final totals	
	8. Termination	


Control Sales Report Module A000
Process Detail Record Module B000
Process Customer Change Module B010
Process Salesman Change Module B020
Process Branch Change Module B030
Print Final Totals Module B040

IPO Chart - Process Detail Records Module

Input/Processing/Output Chart

DATE: June 17	**PROGRAMMER:** Jane Austin	**PROGRAM:** Sales Report
MODULE FUNCTION: Process the Detail Record	**REF:** B000	**MODULE NAME:** Process Detail Record

INPUT	PROCESSING	OUTPUT
1. Sales Input Record	1. Print headings when required	1. Detail Print Line
2. Accumulators	2. Format detail line	2. Updated
a. Customer Total	3. Update accumulators	Accumulators
b. Salesman Total	a. Customer total	a. Customer Total
c. Branch Total	b. Salesman total	b. Salesman Total
d. Final Total	c. Branch total	c. Branch Total
	d. Final total	d. Final Total
	4. Write the detail line	

IPO Chart - Process Customer Change Module

Input/Processing/Output Chart

DATE: June 17	**PROGRAMMER:** Jane Austin	**PROGRAM:** Sales Report
MODULE FUNCTION: Process Customer Change	**REF:** B010	**MODULE NAME:** Process Customer Change

INPUT	PROCESSING	OUTPUT
1. New Customer	1. Format customer total line	1. Customer Total
Number	2. Print the customer total line	Line
2. Customer Total	3. Reset customer total	2. Customer Total
Accumulator	accumulator to zero	Accumulator
	4. Reset customer number	3. Customer Number
	compare area to value of new	Compare Area
	customer key	

234

TM 7

IPO Chart - Process Salesman Change Module

Input/Processing/Output Chart

DATE: June 17
PROGRAMMER: Jane Austin
PROGRAM: Sales Report
MODULE FUNCTION: Process Salesman Change
REF: B020
MODULE NAME: Process Salesman Change

INPUT	PROCESSING	OUTPUT
1. New Salesman	1. Format salesman total line	1. Salesman Total
Number	2. Print the salesman total line	Line
2. Salesman Total	3. Reset salesman total	2. Salesman Total
Accumulator	accumulator to zero	Accumulator
	4. Reset salesman number	3. Salesman Number
	compare area to value of new	Compare Area
	salesman key	

235

253

255

IPO Chart - Process Branch Change Module

Input/Processing/Output Chart

DATE: June 17	PROGRAMMER: Jane Austin	PROGRAM: Sales Report
MODULE FUNCTION: Process Branch Change	REF: B030	MODULE NAME: Process Branch Change

INPUT	PROCESSING	OUTPUT
1. New Branch	1. Format branch total line	1. Branch Total Line
Number	2. Print the branch total line	2. Branch Total
2. Branch Total	3. Reset the branch total	Accumulator
Accumulator	accumulator to zero	3. Branch Number
	4. Reset the branch number	Compare Area
	compare area to the new branch	
	number	

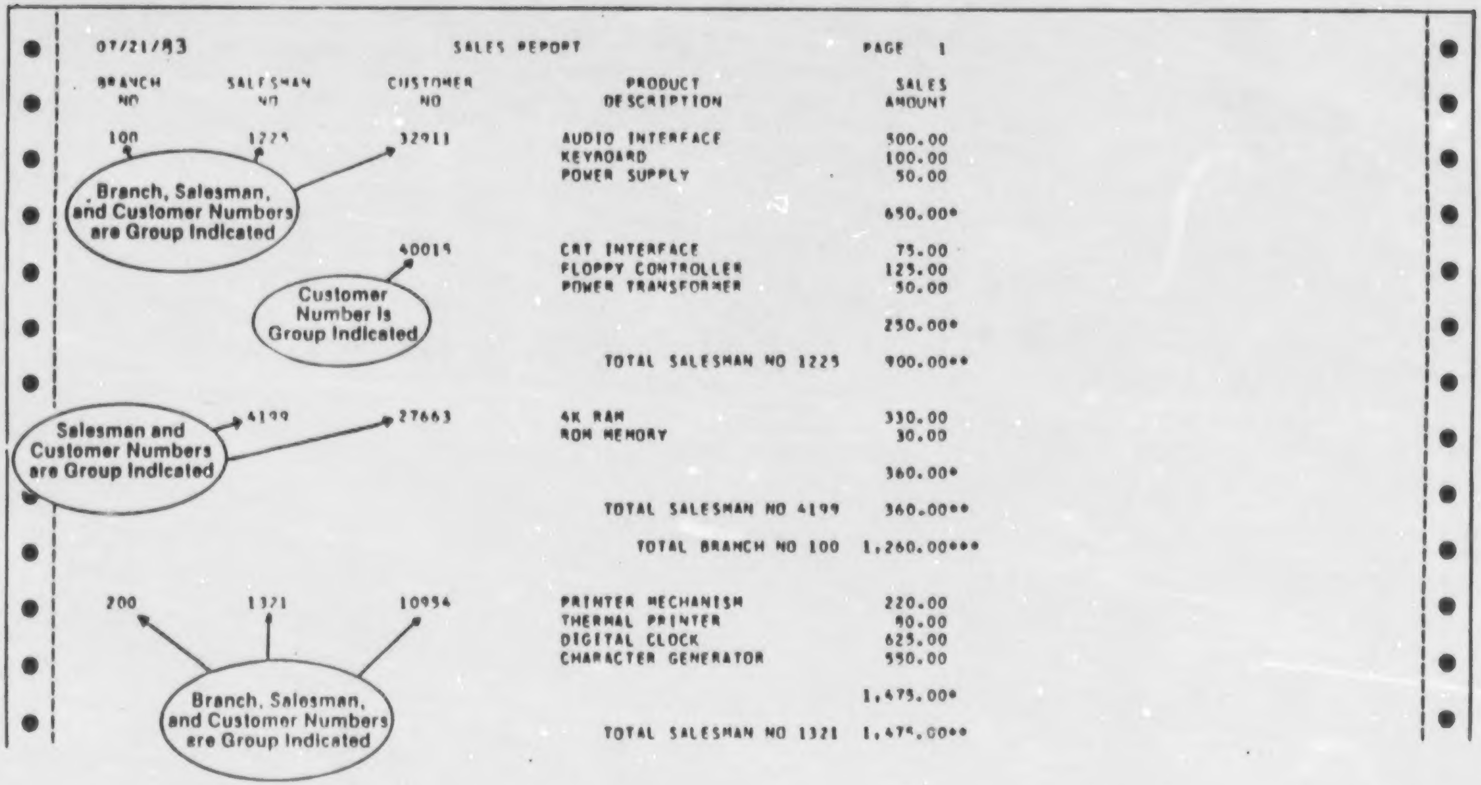
IPO Chart - Print the Final Total Module

Input/Processing/Output Chart

DATE: June 17	**PROGRAMMER:** Jane Austin	**PROGRAM:** Sales Report
MODULE FUNCTION: Print the Final Total	**REF:** B040	**MODULE NAME:** Print Final Total

INPUT	PROCESSING	OUTPUT
1. Final Total	1. Format final total line	1. Final Total Line
Accumulators	2. Write the final total line	

Group-Indicated Report Sample



```
07/21/83                              SALES REPORT                     PAGE   1

BRANCH        SALESMAN       CUSTOMER           PRODUCT                  SALES
  NO             NO             NO            DESCRIPTION                AMOUNT

 100            1225          32911        AUDIO INTERFACE              500.00
                                           KEYBOARD                     100.00
                                           POWER SUPPLY                  50.00

                                                                        650.00*

                              40015        CRT INTERFACE                 75.00
                                           FLOPPY CONTROLLER            125.00
                                           POWER TRANSFORMER             50.00

                                                                        250.00*

                                              TOTAL SALESMAN NO 1225    900.00**

                4199          27663        4K RAM                       330.00
                                           ROM MEMORY                    30.00

                                                                        360.00*

                                              TOTAL SALESMAN NO 4199    360.00**

                                                TOTAL BRANCH NO 100   1,260.00***

 200            1321          10954        PRINTER MECHANISM            220.00
                                           THERMAL PRINTER               80.00
                                           DIGITAL CLOCK                625.00
                                           CHARACTER GENERATOR          550.00

                                                                      1,475.00*

                                              TOTAL SALESMAN NO 1321  1,475.00**
```

Branch, Salesman, and Customer Numbers are Group Indicated

Customer Number Is Group Indicated

Salesman and Customer Numbers are Group Indicated

Branch, Salesman, and Customer Numbers are Group Indicated

COBOL Coding - Control Sales Report Module

```
010160 A000-CONTROL-SALES-REPORT.
010170
010180     OPEN INPUT  SALES-INPUT-FILE
010190          OUTPUT SALES-REPORT-FILE.
010200     READ SALES-INPUT-FILE
011010             AT END
011020                 MOVE 'NO ' TO ARE-THERE-MORE-RECORDS.
011030     IF THERE-IS-A-RECORD
011040         MOVE CUSTOMER-NO-INPUT TO PREVIOUS-CUSTOMER-NUMBER
011050         MOVE SALESMAN-NO-INPUT TO PREVIOUS-SALESMAN-MUMBER
011060         MOVE BRANCH-NO-INPUT TO PREVIOUS-BRANCH-NUMBER
011070         PERFORM A001-PROCESS-AND-READ
011080                 UNTIL THERE-ARE-NO-MORE-RECORDS
011090         PERFORM B010-PROCESS-CUSTOMER-CHANGE
011100         PERFORM B020-PROCESS-SALESMAN-CHANGE
011110         PERFORM B030-PROCESS-BRANCH-CHANGE
011120         PERFORM B040-PRINT-FINAL-TOTAL.
011130     CLOSE SALES-INPUT-FILE
011140           SALES-REPORT-FILE.
011150     STOP RUN.
011160
011170
011180
011190 A001-PROCESS-AND-READ.
011200
012010     IF BRANCH-NO-INPUT NOT = PREVIOUS-BRANCH-NUMBER
012020         PERFORM B010-PROCESS-CUSTOMER-CHANGE
012030         PERFORM B020-PROCESS-SALESMAN-CHANGE
012040         PERFORM B030-PROCESS-BRANCH-CHANGE
012050         MOVE 'YES' TO WAS-THERE-A-BRANCH-CHANGE
012060     ELSE
012070         IF SALESMAN-NO-INPUT NOT = PREVIOUS-SALESMAN-NUMBER
012080             PERFORM B010-PROCESS-CUSTOMER-CHANGE
012090             PERFORM B020-PROCESS-SALESMAN-CHANGE
012100             MOVE 'YES' TO WAS-THERE-A-SALESMAN-CHANGE
012110         ELSE
012120             IF CUSTOMER-NO-INPUT NOT = PREVIOUS-CUSTOMER-NUMBER
012130                 PERFORM B010-PROCESS-CUSTOMER-CHANGE
012140                 MOVE 'YES' TO WAS-THERE-A-CUSTOMER-CHANGE.
012150     PERFORM B000-PROCESS-DETAIL-RECORD.
012160     READ SALES-INPUT-FILE
012170             AT END
012180                 MOVE 'NO ' TO ARE-THERE-MORE-RECORDS.
```

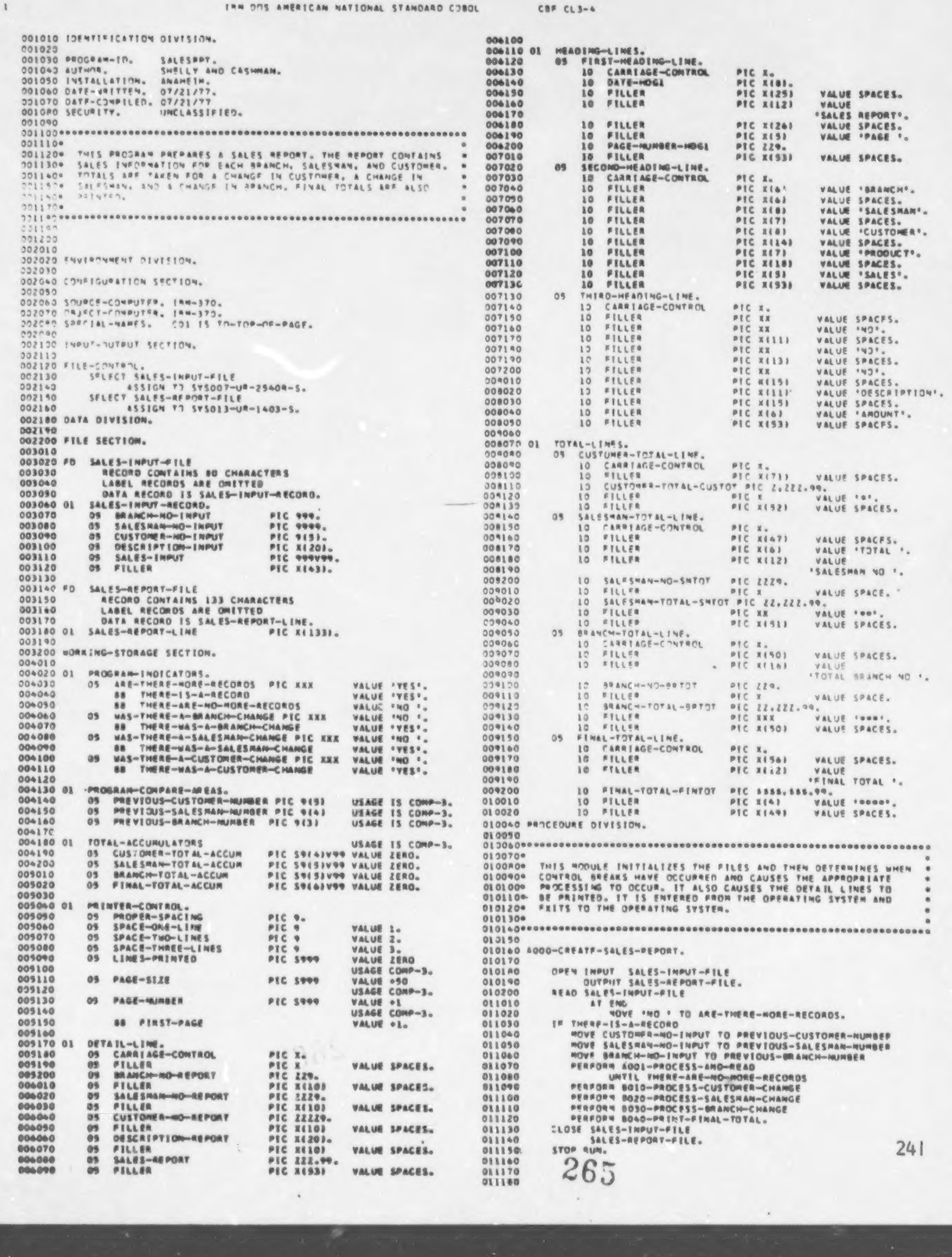
BEST COPY AVAILABLE

COBOL Coding - Process Detail Record Module

```
013100 B000-PROCESS-DETAIL-RECORD.
013110
013120     IF LINES-PRINTED IS EQUAL TO PAGE-SIZE OR
013130             IS GREATER THAN PAGE-SIZE OR
013140             FIRST-PAGE
013150         PERFORM C000-PRINT-HEADINGS
013160         MOVE PREVIOUS-BRANCH-NUMBER TO BRANCH-NO-REPORT
013170         MOVE PREVIOUS-SALESMAN-NUMBER TO SALESMAN-NO-REPORT
013180         MOVE PREVIOUS-CUSTOMER-NUMBER TO CUSTOMER-NO-REPORT.
013190     IF THERE-WAS-A-BRANCH-CHANGE
013200         MOVE BRANCH-NO-INPUT TO BRANCH-NO-REPORT
014010         MOVE SALESMAN-NO-INPUT TO SALESMAN-NO-REPORT
014020         MOVE CUSTOMER-NO-INPUT TO CUSTOMER-NO-REPORT
014030         MOVE 'NO ' TO WAS-THERE-A-BRANCH-CHANGE
014040     ELSE
014050         IF THERE-WAS-A-SALESMAN-CHANGE
014060             MOVE SALESMAN-NO-INPUT TO SALESMAN-NO-REPORT
014070             MOVE CUSTOMER-NO-INPUT TO CUSTOMER-NO-REPORT
014080             MOVE 'NO ' TO WAS-THERE-A-SALESMAN-CHANGE
014090         ELSE
014100             IF THERE-WAS-A-CUSTOMER-CHANGE
014110                 MOVE CUSTOMER-NO-INPUT TO CUSTOMER-NO-REPORT
014120                 MOVE 'NO ' TO WAS-THERE-A-CUSTOMER-CHANGE.
014130     MOVE DESCRIPTIVE-INPUT TO DESCRIPTION-REPORT.
014140     MOVE SALES-INPUT TO SALES-REPORT.
014150     ADD SALES-INPUT TO CUSTOMER-TOTAL-ACCUM
014160                        SALESMAN-TOTAL-ACCUM
014170                        BRANCH-TOTAL-ACCUM
014180                        FINAL-TOTAL-ACCUM.
014190     WRITE SALES-REPORT-LINE FROM DETAIL-LINE
014200             AFTER PROPER-SPACING.
015010     ADD PROPER-SPACING TO LINES-PRINTED.
015020     MOVE SPACE-ONE-LINE TO PROPER-SPACING.
015030     MOVE SPACES TO DETAIL-LINE.
```

BEST COPY AVAILABLE

Source Listing (Part 1)



```
1                      IBM DOS AMERICAN NATIONAL STANDARD COBOL          CBF CL3-4

001010 IDENTIFICATION DIVISION.
001020
001030 PROGRAM-ID.     SALESRPT.
001040 AUTHOR.         SHELLY AND CASHMAN.
001050 INSTALLATION.   ANAHEIM.
001060 DATE-WRITTEN.   07/21/77.
001070 DATE-COMPILED.  07/21/77
001080 SECURITY.       UNCLASSIFIED.
001090
001100*****************************************************************
001110*                                                               *
001120*  THIS PROGRAM PREPARES A SALES REPORT. THE REPORT CONTAINS    *
001130*  SALES INFORMATION FOR EACH BRANCH, SALESMAN, AND CUSTOMER.   *
001140*  TOTALS ARE TAKEN FOR A CHANGE IN CUSTOMER, A CHANGE IN       *
001150*  SALESMAN, AND A CHANGE IN BRANCH. FINAL TOTALS ARE ALSO      *
001160*  PRINTED.                                                     *
001170*                                                               *
001180*****************************************************************
001190
001200
002010
002020 ENVIRONMENT DIVISION.
002030
002040 CONFIGURATION SECTION.
002050
002060 SOURCE-COMPUTER. IBM-370.
002070 OBJECT-COMPUTER. IBM-370.
002080 SPECIAL-NAMES.   C01 IS TO-TOP-OF-PAGE.
002090
002100 INPUT-OUTPUT SECTION.
002110
002120 FILE-CONTROL.
002130     SELECT SALES-INPUT-FILE
002140         ASSIGN TO SYS007-UR-2540R-S.
002150     SELECT SALES-REPORT-FILE
002160         ASSIGN TO SYS013-UR-1403-S.
002180 DATA DIVISION.
002190
002200 FILE SECTION.
003010
003020 FD  SALES-INPUT-FILE
003030       RECORD CONTAINS 80 CHARACTERS
003040       LABEL RECORDS ARE OMITTED
003050       DATA RECORD IS SALES-INPUT-RECORD.
003060 01  SALES-INPUT-RECORD.
003070     05  BRANCH-NO-INPUT         PIC 999.
003080     05  SALESMAN-NO-INPUT       PIC 9999.
003090     05  CUSTOMER-NO-INPUT       PIC 9(5).
003100     05  DESCRIPTION-INPUT       PIC X(20).
003110     05  SALES-INPUT             PIC 999V99.
003120     05  FILLER                  PIC X(43).
003130
003140 FD  SALES-REPORT-FILE
003150       RECORD CONTAINS 133 CHARACTERS
003160       LABEL RECORDS ARE OMITTED
003170       DATA RECORD IS SALES-REPORT-LINE.
003180 01  SALES-REPORT-LINE           PIC X(133).
003190
003200 WORKING-STORAGE SECTION.
004010
004020 01  PROGRAM-INDICATORS.
004030     05  ARE-THERE-MORE-RECORDS  PIC XXX      VALUE 'YES'.
004040         88  THERE-IS-A-RECORD                VALUE 'YES'.
004050         88  THERE-ARE-NO-MORE-RECORDS        VALUE 'NO '.
004060     05  WAS-THERE-A-BRANCH-CHANGE PIC XXX    VALUE 'NO '.
004070         88  THERE-WAS-A-BRANCH-CHANGE        VALUE 'YES'.
004080     05  WAS-THERE-A-SALESMAN-CHANGE PIC XXX  VALUE 'NO '.
004090         88  THERE-WAS-A-SALESMAN-CHANGE      VALUE 'YES'.
004100     05  WAS-THERE-A-CUSTOMER-CHANGE PIC XXX  VALUE 'NO '.
004110         88  THERE-WAS-A-CUSTOMER-CHANGE      VALUE 'YES'.
004120
004130 01  PROGRAM-COMPARE-AREAS.
004140     05  PREVIOUS-CUSTOMER-NUMBER PIC 9(5)    USAGE IS COMP-3.
004150     05  PREVIOUS-SALESMAN-NUMBER PIC 9(4)    USAGE IS COMP-3.
004160     05  PREVIOUS-BRANCH-NUMBER  PIC 9(3)     USAGE IS COMP-3.
004170
004180 01  TOTAL-ACCUMULATORS                       USAGE IS COMP-3.
004190     05  CUSTOMER-TOTAL-ACCUM    PIC S9(4)V99 VALUE ZERO.
004200     05  SALESMAN-TOTAL-ACCUM    PIC S9(5)V99 VALUE ZERO.
005010     05  BRANCH-TOTAL-ACCUM      PIC S9(5)V99 VALUE ZERO.
005020     05  FINAL-TOTAL-ACCUM       PIC S9(6)V99 VALUE ZERO.
005030
005040 01  PRINTER-CONTROL.
005050     05  PROPER-SPACING          PIC 9.
005060     05  SPACE-ONE-LINE          PIC 9        VALUE 1.
005070     05  SPACE-TWO-LINES         PIC 9        VALUE 2.
005080     05  SPACE-THREE-LINES       PIC 9        VALUE 3.
005090     05  LINES-PRINTED           PIC S999     VALUE ZERO
005100                                              USAGE COMP-3.
005110     05  PAGE-SIZE               PIC S999     VALUE +50
005120                                              USAGE COMP-3.
005130     05  PAGE-NUMBER             PIC S999     VALUE +1
005140                                              USAGE COMP-3.
005150         88  FIRST-PAGE                       VALUE +1.
005160
005170 01  DETAIL-LINE.
005180     05  CARRIAGE-CONTROL        PIC X.
005190     05  FILLER                  PIC X        VALUE SPACES.
005200     05  BRANCH-NO-REPORT        PIC ZZ9.
006010     05  FILLER                  PIC X(10)    VALUE SPACES.
006020     05  SALESMAN-NO-REPORT      PIC ZZZ9.
006030     05  FILLER                  PIC X(10)    VALUE SPACES.
006040     05  CUSTOMER-NO-REPORT      PIC ZZZZ9.
006050     05  FILLER                  PIC X(10)    VALUE SPACES.
006060     05  DESCRIPTION-REPORT      PIC X(20).
006070     05  FILLER                  PIC X(10)    VALUE SPACES.
006080     05  SALES-REPORT            PIC ZZZ.99.
006090     05  FILLER                  PIC X(93)    VALUE SPACES.
006100
006110 01  HEADING-LINES.
006120     05  FIRST-HEADING-LINE.
006130         10  CARRIAGE-CONTROL    PIC X.
006140         10  DATE-HDG1           PIC X(8).
006150         10  FILLER              PIC X(25)    VALUE SPACES.
006160         10  FILLER              PIC X(12)    VALUE
006170                                             'SALES REPORT'.
006180         10  FILLER              PIC X(26)    VALUE SPACES.
006190         10  FILLER              PIC X(5)     VALUE 'PAGE '.
006200         10  PAGE-NUMBER-HDG1    PIC ZZ9.
007010         10  FILLER              PIC X(53)    VALUE SPACES.
007020     05  SECOND-HEADING-LINE.
007030         10  CARRIAGE-CONTROL    PIC X.
007040         10  FILLER              PIC X(6)     VALUE 'BRANCH'.
007050         10  FILLER              PIC X(6)     VALUE SPACES.
007060         10  FILLER              PIC X(8)     VALUE 'SALESMAN'.
007070         10  FILLER              PIC X(7)     VALUE SPACES.
007080         10  FILLER              PIC X(8)     VALUE 'CUSTOMER'.
007090         10  FILLER              PIC X(14)    VALUE SPACES.
007100         10  FILLER              PIC X(7)     VALUE 'PRODUCT'.
007110         10  FILLER              PIC X(18)    VALUE SPACES.
007120         10  FILLER              PIC X(5)     VALUE 'SALES'.
007130         10  FILLER              PIC X(53)    VALUE SPACES.
007130     05  THIRD-HEADING-LINE.
007140         10  CARRIAGE-CONTROL    PIC X.
007150         10  FILLER              PIC XX       VALUE SPACES.
007160         10  FILLER              PIC XX       VALUE 'NO'.
007170         10  FILLER              PIC X(11)    VALUE SPACES.
007180         10  FILLER              PIC XX       VALUE 'NO'.
007190         10  FILLER              PIC X(13)    VALUE SPACES.
007200         10  FILLER              PIC XX       VALUE 'NO'.
008010         10  FILLER              PIC X(15)    VALUE SPACES.
008020         10  FILLER              PIC X(11)    VALUE 'DESCRIPTION'.
008030         10  FILLER              PIC X(15)    VALUE SPACES.
008040         10  FILLER              PIC X(6)     VALUE 'AMOUNT'.
008050         10  FILLER              PIC X(53)    VALUE SPACES.
008060
008070 01  TOTAL-LINES.
008080     05  CUSTOMER-TOTAL-LINE.
008090         10  CARRIAGE-CONTROL    PIC X.
008100         10  FILLER              PIC X(71)    VALUE SPACES.
008110         10  CUSTOMER-TOTAL-CUSTOT PIC Z,ZZZ.99.
008120         10  FILLER              PIC X        VALUE '*'.
008130         10  FILLER              PIC X(52)    VALUE SPACES.
008140     05  SALESMAN-TOTAL-LINE.
008150         10  CARRIAGE-CONTROL    PIC X.
008160         10  FILLER              PIC X(47)    VALUE SPACES.
008170         10  FILLER              PIC X(6)     VALUE 'TOTAL '.
008180         10  FILLER              PIC X(12)    VALUE
008190                                             'SALESMAN NO '.
008200         10  SALESMAN-NO-SMTOT   PIC ZZZ9.
009010         10  FILLER              PIC X        VALUE SPACE.
009020         10  SALESMAN-TOTAL-SMTOT PIC ZZ,ZZZ.99.
009030         10  FILLER              PIC XX       VALUE '**'.
009040         10  FILLER              PIC X(51)    VALUE SPACES.
009050     05  BRANCH-TOTAL-LINE.
009060         10  CARRIAGE-CONTROL    PIC X.
009070         10  FILLER              PIC X(50)    VALUE SPACES.
009080         10  FILLER              PIC X(16)    VALUE
009090                                             'TOTAL BRANCH NO '.
009100         10  BRANCH-NO-BRTOT     PIC ZZ9.
009110         10  FILLER              PIC X        VALUE SPACE.
009120         10  BRANCH-TOTAL-BRTOT  PIC ZZ,ZZZ.99.
009130         10  FILLER              PIC XXX      VALUE '***'.
009140         10  FILLER              PIC X(50)    VALUE SPACES.
009150     05  FINAL-TOTAL-LINE.
009160         10  CARRIAGE-CONTROL    PIC X.
009170         10  FILLER              PIC X(56)    VALUE SPACES.
009180         10  FILLER              PIC X(12)    VALUE
009190                                             'FINAL TOTAL '.
009200         10  FINAL-TOTAL-FINTOT  PIC $$$$,$$$.99.
010010         10  FILLER              PIC X(4)     VALUE '****'.
010020         10  FILLER              PIC X(49)    VALUE SPACES.
010040 PROCEDURE DIVISION.
010050
010060*****************************************************************
010070*                                                               *
010080*  THIS MODULE INITIALIZES THE FILES AND THEN DETERMINES WHEN   *
010090*  CONTROL BREAKS HAVE OCCURRED AND CAUSES THE APPROPRIATE      *
010100*  PROCESSING TO OCCUR. IT ALSO CAUSES THE DETAIL LINES TO      *
010110*  BE PRINTED. IT IS ENTERED FROM THE OPERATING SYSTEM AND      *
010120*  EXITS TO THE OPERATING SYSTEM.                               *
010130*                                                               *
010140*****************************************************************
010150
010160 A000-CREATE-SALES-REPORT.
010170
010180     OPEN INPUT  SALES-INPUT-FILE
010190          OUTPUT SALES-REPORT-FILE.
010200     READ SALES-INPUT-FILE
011010         AT END
011020             MOVE 'NO ' TO ARE-THERE-MORE-RECORDS.
011030     IF THERE-IS-A-RECORD
011040         MOVE CUSTOMER-NO-INPUT TO PREVIOUS-CUSTOMER-NUMBER
011050         MOVE SALESMAN-NO-INPUT TO PREVIOUS-SALESMAN-NUMBER
011060         MOVE BRANCH-NO-INPUT TO PREVIOUS-BRANCH-NUMBER
011070         PERFORM A001-PROCESS-AND-READ
011080             UNTIL THERE-ARE-NO-MORE-RECORDS
011090         PERFORM B010-PROCESS-CUSTOMER-CHANGE
011100         PERFORM B020-PROCESS-SALESMAN-CHANGE
011110         PERFORM B030-PROCESS-BRANCH-CHANGE
011120         PERFORM B040-PRINT-FINAL-TOTAL.
011130     CLOSE SALES-INPUT-FILE
011140           SALES-REPORT-FILE.
011150     STOP RUN.
011160
011170
011180
```

BEST COPY AVAILABLE

Source Listing (Part 2)

```
011190 A001-PROCESS-AND-READ.
011200
012010     IF BRANCH-NO-INPUT NOT = PREVIOUS-BRANCH-NUMBER
012020        PERFORM B010-PROCESS-CUSTOMER-CHANGE
012030        PERFORM B020-PROCESS-SALESMAN-CHANGE
012040        PERFORM B030-PROCESS-BRANCH-CHANGE
012050        MOVE 'YES' TO WAS-THERE-A-BRANCH-CHANGE
012060     ELSE
012070        IF SALESMAN-NO-INPUT NOT = PREVIOUS-SALESMAN-NUMBER
012080           PERFORM B010-PROCESS-CUSTOMER-CHANGE
012090           PERFORM B020-PROCESS-SALESMAN-CHANGE
012100           MOVE 'YES' TO WAS-THERE-A-SALESMAN-CHANGE
012110        ELSE
012120           IF CUSTOMER-NO-INPUT NOT = PREVIOUS-CUSTOMER-NUMBER
012130              PERFORM B010-PROCESS-CUSTOMER-CHANGE
012140              MOVE 'YES' TO WAS-THERE-A-CUSTOMER-CHANGE.
012150     PERFORM B000-PROCESS-DETAIL-RECORDS.
012160     READ SALES-INPUT-FILE
012170         AT END
012180             MOVE 'NO ' TO ARE-THERE-MORE-RECORDS.
012200****************************************************************
013010*                                                               *
013020*  THIS MODULE IS ENTERED TO PRINT THE DETAIL LINE FOR THE      *
013030*  REPORT.  IF NECESSARY, IT CAUSES THE HEADINGS TO BE PRINTED  *
013040*  AND THEN FORMATS AND PRINTS THE DETAIL LINE. TOTALS ARE ALSO *
013050*  ACCUMULATED.  THIS MODULE IS ENTERED FROM THE                *
013060*  A000-CREATE-SALES-REPORT MODULE AND EXITS BACK TO IT.        *
013070*                                                               *
013080****************************************************************
013090
013100 B000-PROCESS-DETAIL-RECORDS.
013110
013120     IF LINES-PRINTED IS EQUAL TO PAGE-SIZE OR
013130           IS GREATER THAN PAGE-SIZE OR
013140           FIRST-PAGE
013150        PERFORM C000-PRINT-HEADINGS
013160        MOVE PREVIOUS-BRANCH-NUMBER TO BRANCH-NO-REPORT
013170        MOVE PREVIOUS-SALESMAN-NUMBER TO SALESMAN-NO-REPORT
013180        MOVE PREVIOUS-CUSTOMER-NUMBER TO CUSTOMER-NO-REPORT.
013190     IF THERE-WAS-A-BRANCH-CHANGE
013200        MOVE BRANCH-NO-INPUT TO BRANCH-NO-REPORT
014010        MOVE SALESMAN-NO-INPUT TO SALESMAN-NO-REPORT
014020        MOVE CUSTOMER-NO-INPUT TO CUSTOMER-NO-REPORT
014030        MOVE 'NO ' TO WAS-THERE-A-BRANCH-CHANGE
014040     ELSE
014050        IF THERE-WAS-A-SALESMAN-CHANGE
014060           MOVE SALESMAN-NO-INPUT TO SALESMAN-NO-REPORT
014070           MOVE CUSTOMER-NO-INPUT TO CUSTOMER-NO-REPORT
014080           MOVE 'NO ' TO WAS-THERE-A-SALESMAN-CHANGE
014090        ELSE
014100           IF THERE-WAS-A-CUSTOMER-CHANGE
014110              MOVE CUSTOMER-NO-INPUT TO CUSTOMER-NO-REPORT
014120              MOVE 'NO ' TO WAS-THERE-A-CUSTOMER-CHANGE.
014130     MOVE DESCRIPTION-INPUT TO DESCRIPTION-REPORT.
014140     MOVE SALES-INPUT TO SALES-REPORT.
014150     ADD SALES-INPUT TO CUSTOMER-TOTAL-ACCUM
014160                        SALESMAN-TOTAL-ACCUM
014170                        BRANCH-TOTAL-ACCUM
014180                        FINAL-TOTAL-ACCUM.
014190     WRITE SALES-REPORT-LINE FROM DETAIL-LINE
014200         AFTER PROPER-SPACING.
015010     ADD PROPER-SPACING TO LINES-PRINTED.
015020     MOVE SPACE-ONE-LINE TO PROPER-SPACING.
015030     MOVE SPACES TO DETAIL-LINE.
015040****************************************************************
015050*                                                               *
015060*  THIS MODULE IS ENTERED TO PROCESS A CHANGE IN CUSTOMER       *
015070*  NUMBER.  IT PRINTS THE CUSTOMER TOTAL AND RESETS THE         *
015080*  COMPARE AREA AND COUNTER.  IT IS ENTERED FROM THE            *
015090*  A000-CREATE-SALES-REPORT MODULE AND EXITS BACK TO IT.        *
015100*                                                               *
015110****************************************************************
015120
015130 B010-PROCESS-CUSTOMER-CHANGE.
015140
015150     MOVE CUSTOMER-TOTAL-ACCUM TO CUSTOMER-TOTAL-CUSTOT.
015160     WRITE SALES-REPORT-LINE FROM CUSTOMER-TOTAL-LINE
015170         AFTER ADVANCING 2 LINES.
015180     MOVE ZEROS TO CUSTOMER-TOTAL-ACCUM
015190     MOVE CUSTOMER-NO-INPUT TO PREVIOUS-CUSTOMER-NUMBER.
015200     ADD 2 TO LINES-PRINTED.
016010     MOVE SPACE-TWO-LINES TO PROPER-SPACING.
016030****************************************************************
016040*                                                               *
016050*  THIS MODULE IS ENTERED TO PROCESS A CHANGE IN SALESMAN       *
016060*  NUMBER.  IT PRINTS THE SALESMAN TOTAL AND RESETS THE         *
016070*  COMPARE AREA AND COUNTER.  IT IS ENTERED FROM THE            *
016080*  A000-CREATE-SALES-REPORT MODULE AND EXITS TO IT.             *
016090*                                                               *
016100****************************************************************
016110
016120 B020-PROCESS-SALESMAN-CHANGE.
016130
016140     MOVE PREVIOUS-SALESMAN-NUMBER TO SALESMAN-NO-SMTOT.
016150     MOVE SALESMAN-TOTAL-ACCUM TO SALESMAN-TOTAL-SMTOT.
016160     WRITE SALES-REPORT-LINE FROM SALESMAN-TOTAL-LINE
016170         AFTER ADVANCING 2 LINES.
016180     MOVE ZEROS TO SALESMAN-TOTAL-ACCUM.
016190     MOVE SALESMAN-NO-INPUT TO PREVIOUS-SALESMAN-NUMBER.
016200     ADD 2 TO LINES-PRINTED.
017010     MOVE SPACE-THREE-LINES TO PROPER-SPACING.
017030****************************************************************
017040*                                                               *
017050*  THIS MODULE IS ENTERED TO PROCESS A CHANGE IN BRANCH NUMBER.*
017060*  IT PRINTS THE BRANCH TOTAL AND RESETS THE COMPARE AREA AND   *
017070*  COUNTER.  IT IS ENTERED FROM THE A000-CREATE-SALES-REPORT    *
017080*  MODULE AND EXITS BACK TO IT.                                 *
017090*                                                               *
017100****************************************************************
017110
017120 B030-PROCESS-BRANCH-CHANGE.
017130
017140     MOVE PREVIOUS-BRANCH-NUMBER TO BRANCH-NO-BRTOT.
017150     MOVE BRANCH-TOTAL-ACCUM TO BRANCH-TOTAL-BRTOT.
017160     WRITE SALES-REPORT-LINE FROM BRANCH-TOTAL-LINE
017170         AFTER ADVANCING 2 LINES.
017180     MOVE ZEROS TO BRANCH-TOTAL-ACCUM.
017190     MOVE BRANCH-NO-INPUT TO PREVIOUS-BRANCH-NUMBER.
017200     ADD 2 TO LINES-PRINTED.
018010     MOVE SPACE-THREE-LINES TO PROPER-SPACING.
018030****************************************************************
018040*                                                               *
018050*  THIS MODULE IS ENTERED TO PROCESS THE FINAL TOTAL.  THE      *
018060*  FINAL TOTAL IS MOVED TO THE OUTPUT AREA AND PRINTED.  THIS   *
018070*  MODULE IS ENTERED FROM THE A000-CREATE-SALES-REPORT MODULE   *
018080*  AND EXITS BACK TO IT.                                        *
018090*                                                               *
018100****************************************************************
018110
018120 B040-PRINT-FINAL-TOTAL.
018130
018140     MOVE FINAL-TOTAL-ACCUM TO FINAL-TOTAL-FINTOT.
018150     WRITE SALES-REPORT-LINE FROM FINAL-TOTAL-LINE
018160         AFTER ADVANCING 3 LINES.
018190****************************************************************
018200*                                                               *
019010*  THIS MODULE PRINTS THE HEADINGS ON THE REPORT.  IT IS        *
019020*  ENTERED FROM THE B000-PROCESS-DETAIL-RECORDS MODULE AND      *
019030*  EXITS TO THE SAME MODULE.                                    *
019040*                                                               *
019050****************************************************************
019060
019070 C000-PRINT-HEADINGS.
019080
019090     MOVE CURRENT-DATE TO DATE-HDG1.
019100     MOVE PAGE-NUMBER TO PAGE-NUMBER-HDG1.
019110     WRITE SALES-REPORT-LINE FROM FIRST-HEADING-LINE
019120         AFTER ADVANCING TO-TOP-OF-PAGE.
019130     ADD 1 TO PAGE-NUMBER.
019140     WRITE SALES-REPORT-LINE FROM SECOND-HEADING-LINE
019150         AFTER ADVANCING 2 LINES.
019160     WRITE SALES-REPORT-LINE FROM THIRD-HEADING-LINE
019170         AFTER ADVANCING 1 LINES.
019180     MOVE SPACE-TWO-LINES TO PROPER-SPACING.
019190     MOVE ZERO TO LINES-PRINTED.
```

MULTIPLE-LEVEL CONTROL BREAKS

UNIT VII

ASSIGNMENT SHEET 1: Terms and Introduction to Multiple-Level Control Breaks

1. Briefly explain the meaning of the term **multiple-level control break.**

2. What is the name of the driver module for the sample program in this unit?

3. What does **group-indicated** mean?

4. What is a group-indicated item in TM 1?

MULTIPLE-LEVEL CONTROL BREAKS

UNIT VII

ASSIGNMENT SHEET 2: Overview of Sample Program, Initial, and Detailed Processing

1. Briefly summarize the major programming specifications (HO 1).

2. Briefly explain the two steps indicated by an arrow and small-circled number on TM 3.

3. Briefly explain the steps in processing of detailed records.

MULTIPLE-LEVEL CONTROL BREAKS

UNIT VII

ASSIGNMENT SHEET 3: Actual Processing of the Four Levels of Control Breaks

1. For each of the four levels of control breaks, specify
   a. The key
   b. The fields accumulated
   c. The fields zeroed

ASSIGNMENT SHEET 4: IPO Charts for Sample Program Modules

1. Make summarized IPO Charts for each module below from the sample program in the format:

	Module Name	Major Input	Major Processing	Major Outputs
a.	Control Sales Report Module			
b.	Process Detail Record Module			
c.	Process Customer Change Module			
d.	Process Salesman Change Module			

ASSIGNMENT SHEET 5: Pseudocode for Sample Program Modules

Very briefly and in your own language summarize the pseudocode (major items) of HO 2, HO 3, and HO 5.

1. HO 2 - Control Sales Report Module

2. HO 3 - Process Detail Records Module

3. HO 5 - Process Salesman Change Module

MULTIPLE-LEVEL CONTROL BREAKS

UNIT VII

ANSWERS TO ASSIGNMENT SHEETS

Assignment Sheet 1

1. A program with multiple, hierarchical levels of totals; each total is a subset of the next higher total.
2. A 000-CREATE-SALES-REPORT
3. Printing is suppressed, once the data is first printed, for repetitious information.
4. Branch, Salesman, or Customer

Assignment Sheet 2

1. a. The program is to read sales cards and print the Sales Report with headings, one line of printing for each card and breaks on Customer Number, Salesman Number and Branch Number.
   b. The program is to be written in COBOL and input records are to be sorted by Customer Number within Salesman Number within Branch Number.
2. a. Circle number one refers to reading the first sales card and setting up the Compare Area.
   b. Circle number two refers to printing the first detail line from the first sales card.
3. a. Set up output area
   b. Accumulate sales totals
   c. Print the detailed record

Assignment Sheet 3

1. Lowest level
   a. Key - Customer Number
   b. Fields accumulated
      1. Customer Total
      2. Salesman Total
      3. Branch Total
      4. Final Total
   c. Fields zeroed - Customer Total
2. Second lowest level
   a. Key - Salesman Number
   b. Fields accumulated
      1. Salesman Total
      2. Branch Total
      3. Final Total
   c. Fields zeroed
      1. Customer Total
      2. Salesman Total

272

3. Third-lowest or second-highest level
    a. Key - Branch Number
    b. Fields accumulated
        1. Branch Total
        2. Final Total
    c. Fields zeroed
        1. Customer Total
        2. Salesman Total
        3. Branch Total
4. Fourth-lowest or highest level
    a. Key - Final or Company Total
    b. Fields accumulated
        1. Final Total
    c. Fields zeroed - none as end of report or all if another company is assumed possible

Assignment Sheet 4

	Module Name	Major Input	Major Processing	Major Outputs
a.	Control Sales Report Module	Sales Cards	Supervises other modules and detect breaks	Sales Report
b.	Process Detail Record Module	Sales Detail Record	Accumulate four totals	Detailed Line
c.	Process Customer Change Module	New Customer Number	Format Customer Total Line and reset totals	Customer Total Line
d.	Process Salesman Change Module	New Salesman Number	Format Salesman Total Line and reset totals	Salesman Total Line

Assignment Sheet 5

1. a. Detects breaks and performs relevant module for each break; if no break, performs detail record module
    b. Controls recoding of records
    c. Controls processing of final totals
2. Sets up printer output area, accumulates Sales Totals and prints a Detail Line.
3. a. Move Salesman Number and Salesman Total to print line, print Salesman Total Line.
    b. Set up accumulator for next salesman.

MULTIPLE LEVEL CONTROL BREAKS

UNIT VII

EXAMINATION

1. Match each term in Column A with its correct phrase in Column B.

		Column A		Column B
____	a.	Multiple-Level Control Breaks	1.	Controls other modules
____	b.	Driver Module	2.	Will not always be printed
____	c.	Control key	3.	Important for sorts, control breaks, file access, etc.
____	d.	Group indicated	4.	Group of control breaks

2. If a report were being prepared with multiple level control breaks for a multi-divisional company payroll, what would be the likely sequence of control breaks of the following scrambled fields?

____ a. Company

____ b. Employee

____ c. Division

____ d. Department

____ e. Shift

3. What fields are used as control keys for the sample program that was studied in this unit? Which is the most minor key?

4. What fields are used in the compare area of this sample program?

5. List three main steps in processing of detailed records.

a.

b.

c.

6. Explain first-level control break.

7. How is the IPO Chart for the Process Detail Records Module related to the IPO Chart for the Control Sales Report Module?

8. Which module starts the use of the READ-BEFORE-PROCESSING technique?

9. What type of nested IF statement is used in the Process Detailed Records Module?

10. In the sample program studied in this unit, what level of data item was the field coded as "THERE-IS-A-RECORD"?

MULTIPLE-LEVEL CONTROL BREAKS

UNIT VII

ANSWERS TO THE EXAMINATION

1. a. 4
   b. 1
   c. 3
   d. 2

2. 1st level - employee
   2nd level - shift
   3rd level - department
   4th level - division
   5th level - company (or final total level)

3. Fields used are branch, employee, and customer. Customer is most minor.

4. Branch number, employee number, and customer number

5. Set up output area, accumulate sales totals, print detailed record.

6. The first-level control break is on Customer. The totals for Customer are printed, and the Customer Total Accumulator is zeroed.

7. The IPO Chart for the Process Detailed Records Modules is summarized as one line of processing in the IPO Chart for the Control Sales Report Module.

8. The driver module - Control Sales Report Module

9. Matched IF/ELSE statements or exclusive conditions

10. Level 88

ADDITIONAL COBOL ELEMENTS

UNIT VIII

UNIT OBJECTIVE

After studying this unit, the students should be able to apply additional COBOL elements in their programs.

SPECIFIC OBJECTIVES

After studying this unit, students should be able to:

1. Match terms and definitions.
2. Code level 77 entries and maintain programs which use them.
3. Code level 66 entries and maintain programs which use them.
4. Write qualification and maintain programs which use qualifications.
5. Code BLANK WHEN ZERO clause in programs.
6. Write statements utilizing figurative constants in programs.
7. Explain why the use of GO TO's is generally avoided in new programs.
8. Explain why the use of the ALTER statement is generally avoided in new programs.
9. Code ACCEPT and DISPLAY statements for low-volume data in programs.
10. Write an ON SIZE ERROR in arithmetic statements.
11. Explain the function of the INSPECT statement in edit routines.
12. Explain the use of the STRING/UNSTRING statement.

MULTIPLE-LEVEL CONTROL BREAKS

UNIT VIII

SUGGESTED ACTIVITIES

I. Instructor

- A. Make transparencies.
- B. Provide students with objectives and discuss.
- C. Provide students with information and assignment sheets and discuss these sheets.
- D. Make parallel assignments for reading and programming from text.
- E. Provide supplementary information and assignments.
- F. Give examination.

II. Student

- A. Read objectives.
- B. Study information sheets and text assignments.
- C. Complete assignment sheets and supplementary assignments.
- D. Complete examination.

INSTRUCTIONAL MATERIALS

I. Unit content

- A. Objectives
- B. Information sheets
- C. Transparency masters
    - 1. TM 1: Example of Renames Clause
    - 2. TM 2: Example of Name Qualification
    - 3. TM 3: Example of GO TO DEPENDING Statement
    - 4. TM 4: General Formats of INSPECT Statement
    - 5. TM 5: Example of INSPECT Statement (Part 1 and Part 2)

D. Assignment sheets

   1. AS 1: Level 77 and Level 66 Entries

   2. AS 2: Name Qualification, BLANK WHEN ZERO, and Figurative Constants

   3. AS 3: GO TO, ACCEPT/DISPLAY, and ON SIZE ERROR Statements

E. Answers to assignment sheets

F. Examination

G. Answers to examination

II. References

A. Shelly, Gary B. and Thomas J. Cashman. "Introduction to Computer Programming." in Structured COBOL. Fullerton, CA: Anaheim Publishing Company, 1977.

ADDITIONAL COBOL ELEMENTS
UNIT VIII

INFORMATION SHEET

I. Terms and functional definitions

A. **Level 77** - an independent data item; cannot be grouped or subdivided; used for storage of the results of calculations or switches

B. **Collating sequence** - order of sorting

C. **Forward GO TO** - a GO TO which branches forward; normally within the range of a PERFORM THRU

D. **Case Structure** - conditions systematically arrayed as alternative conditions as with GO TO DEPENDING

E. **Operator's Console** - control console for a mainframe computer

F. **Delimiter** - establishes syntactical boundaries, as in COBOL where a space is a delimiter for a word.

G. **Level 66** - used to rename other data definition

H. **BLANK WHEN ZERO** - used to get on output field to spaces when its value is zero

I. **Figurative constants** - constants used figuratively as literals

J. **ACCEPT/DISPLAY** - low-volume, nonfile oriented I/O verbs

K. **ON SIZE ERROR** - conditional statement used to prevent arithmetic overflow

L. **INSPECT** - used to convert characters or count characters; operates on characters rather than fields

M. **STRING/UNSTRING** - string operator verb

II. Level 77 Entries

A. Level 77 is called an independent data item since it stands alone and cannot be subdivided or treated as a group item.

B. Examples:

```
77   END-OF-JOB-SWITCH  PIC X VALUE 'N'
77   TOTAL-SALES   PIC S9(5)V99 VALUE ZERO,
```

C. Generally they are used in older programs as switches or unrelated working totals. More commonly now replaced by 01's grouped with 05's for easier reading by grouping.

D. They are not recommended for new usage as the new standard proposes to drop the item. In maintenance work, replace with 01 levels as necessary.

III. Level 66 entries - renames clause (TM 1)

A. Level 66 data entries are used to rename another data item or group of data items. This clause is used for ease of reference or clarity of documentation.

B. Rules:

1. Level 66 entries must immediately follow the items being renamed

2. Level 66 entries cannot cause restructuring or be used to restructure the data structure. This is the function of the REDEFINES clause. (See discussion of tables in your text.)

C. Level 66 entries are not generally useful for new code but are recommended for maintenance work to improve clarity of older programs

IV. Name qualification with nonunique data names (TM 2)

A. Name qualification is used to make nonunique data names unique by reference to a higher-level group, record, or file name of which the nonunique data must be a part.

B. Rules

1. Every reference to any nonunique data name must be made unique by including a qualifier.

2. Multiple levels of qualification may be used.

C. Advantages

1. Blocks of data definitions can be replicated; e.g., as in the input master file and the output or updated master file, which allows having the same data structure and saving the coding effort to define the data.

2. Whole records can be moved and formatted in a single MOVE CORRESPONDING statement.

3. Helps enforce data definition standards

D. Disadvantages

1. Name qualification is not generally used because data structures must correspond exactly down to the lowest level elementary item. If not, portions of the data may not be moved, causing errors which

are difficult to find. For example, in TM 2 the address would not be moved with a MOVE CORRESPONDING statement.

2. This is usually not recommended for new code. However, the trainee may be expected to maintain older programs occasionally with this feature, so some familiarity is expected.

V. BLANK WHEN ZERO clause

A. Blanks a numeric field on a report when its value is zero

B. Recommended for cleaner report design

VI. The figurative constants

A. Called figurative constants because when used, they are spelled out figuratively; e.g., zero is not 0.0 but "ZERO"

B. Examples of common figurative constants:

QUOTES	for quotes instead of the quote mark
SPACES	for blanks instead of an actual blank
ZERO	for numeric value of zero, must be used with computational data items only
HIGH-VALUES	for highest-collating-sequence value; i.e., sorts last if moved into sort key of a record
LOW-VALUES	for lowest-collating-sequence value, sorts first
ALL	for special case must be used with another literal as in MOVE ALL '6' would move all sixes to a field

C. Usage note: HIGH-VALUES and LOW-VALUES are preferred for use to help decouple the application from a vendor's collating sequence; e.g., do not use the machine's highest-collating-sequence value, such as hexadecimal 'FFFF' but replace this literal with HIGH-VALUES. If the vendor changes, the application need not be recoded by hand.

D. NOTE: All of the figurative constants are smart. Their use will cause the COBOL compiler to examine the intended receiving field and generate sufficient code to fill the field.

```
MOVE SPACE TO BIG-FIELD
```

will generate enough blanks to completely set all the positions in BIG-FIELD to blanks

VII. GO TO formats, usage, and case structure

A. Formats:

Unconditional GO TO

```
1.    GO TO          (Paragraph-name)
                      (Section-name)
```

Conditional

```
2.    GO TO          Paragraph-name-1
                      Paragraph-name-2
```

B. Comments - Format 1

1. Format 1 is generally not recommended for new applications. Many feel that excessive use leads to bad code, resulting in structures which are fragmented and difficult to follow and maintain. However, it is used frequently in older code.

2. Format 1 that is used within the range of a PERFORM THRU is generally recognized as useful. It is sometimes called a forward GO TO.

   Example:

```
PERFORM PARA-A THRU A-EXIT.

PARA-A.

      IF ERROR GOTO A-EXIT

A-EXIT.

      EXIT .
```

3. In the forward GO TO, the GO TO is used to avoid further processing when an error or abnormal condition is recognized.

4. Forward GO TO should only go to the EXIT paragraph, which causes a normal return to the performing module.

5. Forward GO TO should never "jump-out" of the range of the PERFORM/THRU.

C. Comments - Format 2

1. Format 2, called the case structure, is a recommended and useful structure.

2. A case structure is a conditional transfer of control depending upon the occurrence of a predefined value in a control field. (TM 3)

3. The case structure can also be implemented without the GO TO depending on the use of a series of nested, paired IF/ELSE statements.

VIII. The ALTER statement

A. The ALTER statement causes an unconditional GO TO to be selectively changed at execution time so as to change where the ALTER says to go. This can be confusing for the maintenance programmer so it is not recommended.

B. The ALTER statement is generally poor code design.

C. Usage is **not** recommended. Furthermore, the proposed COBOL standard plans to drop support of the ALTER, thereby rendering any program which uses it to be virtually uncompilable. Special effort should be made to remove it from existing code.

IX. ACCEPT and DISPLAY statements

A. Originally used to get (accept) low volume data into the CPU from an operator's console but now increasingly used to put (display) messages on terminals or get (accept) data from CRT's.

B. Not necessary to define files; i.e., not like READ's and WRITE's - a unique I/O verb because it does not need files

C. ACCEPT and DISPLAY statements are now in use on CRT screen terminals as the I/O verb for reading and writing screens. COBOL does not have a standard facility for interfacing with CRT's, so the trainee must consult the vendor's manual for each system.

X. ON SIZE ERROR option

A. This is a control option which, when added to arithmetic operations, prevents loss of data under a condition in which a resultant field could not be stored in the receiving field. Example: division by the value zero results in an infinite value which cannot be stored.

B. Example

```
DIVIDE SUM-OF-THE-PRICES BY TOTAL-SOLD GIVING AVERAGE-

          PRICE ON SIZE ERROR PERFORM DIVISION-ERROR-RTN.
```

C. Rules

1. ON SIZE ERROR is like the AT END on READ statements. It must be ended with a period.
2. ELSE cannot be combined with the ON SIZE ERROR.

XI. INSPECT Statement (TM 4 and 5)

A. The INSPECT statement replaces the older EXAMINE statement used in many ways for character-by-character count and/or to replace characters in a data item

B. The INSPECT statement is used to convert characters from one format to another when changing equipment brands. It is also much used in editing of input data to detect errors or in unstructured formats as in CRT screen input.

C. Many formats are possible. TM 4 illustrates major formats.

D. Usage examples are given in TM 5.

XII. STRING/UNSTRING statement

A. Takes alphanumeric data from two or more fields and places it in a single field (e.g., last name, first name, and middle initial), which is being put together from separately entered fields.

B. UNSTRING is the opposite of STRING; but a delimiter, such as a comma or blank must separate the data portions so that they can be placed into separate fields. The delimiter must be included in the statement.

Example of Renames Clause

$$66\ \text{data-name-1; }\underline{\text{RENAMES}}\ \text{data-name-2}\ \left[\left\{\begin{array}{l}\underline{\text{THROUGH}}\\ \underline{\text{THRU}}\end{array}\right\}\ \text{data-name-3}\right].$$

```
003100 05  NAME-INPUT               PIC X(20).
003110 05  ADDRESS-INPUT.
003120        10  STREET-INPUT      PIC X(15).
003130        10  CITY-INPUT        PIC X(15).
003140        10  STATE-INPUT       PIC XX.
003150        10  ZIP-CODE-INPUT    PIC 99999.
003160 66  NAME-STREET-CITY RENAMES NAME-INPUT THRU CITY-INPUT.
```

Example of Name Qualification

```
003020 01  TRANSACTION-RECORD.
003030     05  NAME-EMPLOYEE.
003040         10  FIRST-NAME          PIC X(10).
003050         10  LAST-NAME           PIC X(15).
003060     05  ADDRESS-EMPLOYEE.
003070         10  STREET-NUMBER       PIC X(5).
003080         10  STREET-NAME         PIC X(10).
003090         CITY-EMPLOYEE           PIC X(20).
003100         STATE-EMPLOYEE          PIC X(15).
003110
003120 01  MASTER-RECORD.
003130     05  NAME-EMPLOYEE.
003140         10  FIRST-NAME          PIC X(10).
003150         10  LAST-NAME           PIC X(15).
003160     05  ADDRESS-EMPLOYEE        PIC X(15).
003170     05  CITY-EMPLOYEE           PIC X(20).
003180     05  STATE-EMPLOYEE          PIC X(15).
```

Example of GO TO DEPENDING Statement

```
GO TO procedure-name-1  [, procedure-name-2] ...   , procedure-name-n
   DEPENDING ON identifier
```

```
018130     GO TO C001-ADD-A-RECORD
018140           C002-DELETE-A-RECORD
018150           C003-CHANGE-SALES-AMOUNT
018160           C004-CHANGE-CUSTOMER-ADDRESS
018170           C005-CHANGE-CREDIT-LIMIT
018180        DEPENDING ON TRANSACTION-CODE-INPUT.
```

TM 4

BEST COPY AVAILABLE

General Formats of INSPECT Statement

Format 1

$$\underline{\text{INSPECT}}\ \text{identifier-1}\ \underline{\text{TALLYING}}$$

$$\left\{ ,\ \text{identifier-2}\ \underline{\text{FOR}} \left\{ ,\ \left\{ \begin{matrix} \begin{Bmatrix} \underline{\text{ALL}} \\ \underline{\text{LEADING}} \end{Bmatrix} \begin{Bmatrix} \text{identifier-3} \\ \text{literal-1} \end{Bmatrix} \\ \underline{\text{CHARACTERS}} \end{matrix} \right\} \left[\begin{Bmatrix} \underline{\text{BEFORE}} \\ \underline{\text{AFTER}} \end{Bmatrix} \ \text{INITIAL}\ \begin{Bmatrix} \text{identifier-4} \\ \text{literal-2} \end{Bmatrix} \right] \right\} \ldots \right\} \ldots$$

Format 2

$$\underline{\text{INSPECT}}\ \text{identifier-1}\ \underline{\text{REPLACING}}$$

$$\left\{ \begin{matrix} \underline{\text{CHARACTERS}}\ \underline{\text{BY}} \begin{Bmatrix} \text{identifier-6} \\ \text{literal-4} \end{Bmatrix} \left[\begin{Bmatrix} \underline{\text{BEFORE}} \\ \underline{\text{AFTER}} \end{Bmatrix} \ \text{INITIAL}\ \begin{Bmatrix} \text{identifier-7} \\ \text{literal-5} \end{Bmatrix} \right] \\ \left\{ ,\ \begin{Bmatrix} \underline{\text{ALL}} \\ \underline{\text{LEADING}} \\ \underline{\text{FIRST}} \end{Bmatrix} \left\{ ,\ \begin{Bmatrix} \text{identifier-5} \\ \text{literal-3} \end{Bmatrix} \ \underline{\text{BY}}\ \begin{Bmatrix} \text{identifier-6} \\ \text{literal-4} \end{Bmatrix} \left[\begin{Bmatrix} \underline{\text{BEFORE}} \\ \underline{\text{AFTER}} \end{Bmatrix} \ \text{INITIAL}\ \begin{Bmatrix} \text{identifier-7} \\ \text{literal-5} \end{Bmatrix} \right] \right\} \ldots \right\} \ldots \end{matrix} \right\}$$

Format 3

$$\underline{\text{INSPECT}}\ \text{identifier-1}\ \underline{\text{TALLYING}}$$

$$\left\{ ,\ \text{identifier-2}\ \underline{\text{FOR}} \left\{ ,\ \left\{ \begin{matrix} \begin{Bmatrix} \underline{\text{ALL}} \\ \underline{\text{LEADING}} \end{Bmatrix} \begin{Bmatrix} \text{identifier-3} \\ \text{literal-1} \end{Bmatrix} \\ \underline{\text{CHARACTERS}} \end{matrix} \right\} \left[\begin{Bmatrix} \underline{\text{BEFORE}} \\ \underline{\text{AFTER}} \end{Bmatrix} \ \text{INITIAL}\ \begin{Bmatrix} \text{identifier-4} \\ \text{literal-2} \end{Bmatrix} \right] \right\} \ldots \right\} \ldots$$

$$\underline{\text{REPLACING}}$$

$$\left\{ \begin{matrix} \underline{\text{CHARACTERS}}\ \underline{\text{BY}} \begin{Bmatrix} \text{identifier-6} \\ \text{literal-4} \end{Bmatrix} \left[\begin{Bmatrix} \underline{\text{BEFORE}} \\ \underline{\text{AFTER}} \end{Bmatrix} \ \text{INITIAL}\ \begin{Bmatrix} \text{identifier-7} \\ \text{literal-5} \end{Bmatrix} \right] \\ \left\{ ,\ \begin{Bmatrix} \underline{\text{ALL}} \\ \underline{\text{LEADING}} \\ \underline{\text{FIRST}} \end{Bmatrix} \left\{ ,\ \begin{Bmatrix} \text{identifier-5} \\ \text{literal-3} \end{Bmatrix} \ \underline{\text{BY}}\ \begin{Bmatrix} \text{identifier-6} \\ \text{literal-4} \end{Bmatrix} \left[\begin{Bmatrix} \underline{\text{BEFORE}} \\ \underline{\text{AFTER}} \end{Bmatrix} \ \text{INITIAL}\ \begin{Bmatrix} \text{identifier-7} \\ \text{literal-5} \end{Bmatrix} \right] \right\} \ldots \right\} \ldots \end{matrix} \right\}$$

Example of INSPECT Statement
(Part 1 of 2)

```
017150      INSPECT ADDRESS-INPUT
017160          TALLYING BLANK-COUNT FOR ALL ' '.
```

ADDRESS-INPUT

1	2	7		S	M	I	T	H		A	V	E		

BLANK-COUNT
(After Execution)

0	0	4

```
019180      INSPECT NAME-INPUT
019190          TALLYING A-COUNT FOR ALL 'A' BEFORE INITIAL ' '.
```

NAME-INPUT

A	R	M	O	N	A		K	.		J	O	H	N	S

A-COUNT
(After Execution)

0	0	2

Example of INSPECT Statement
(Part 2 of 2)

```
019150     INSPECT AMOUNT-INPUT
019160         REPLACING LEADING ' ' BY '0'.
```

	Before Execution		After Execution
AMOUNT-INPUT	␣ ␣ ␣ ␣ 2 3 0 8	**AMOUNT-INPUT**	0 0 0 0 2 3 0 8

```
029140     INSPECT DATE-FIELD
029150         REPLACING ALL ' ' BY '-'.
```

DATE-FIELD	0 9 ␣ 2 7 ␣ 7 9	**DATE-FIELD**	0 9 - 2 7 - 7 9

ADDITIONAL COBOL ELEMENTS

UNIT VIII

ASSIGNMENT SHEET 1: Level 77 and Level 66 Entries

1. Convert the following Level 77 entries to 01's.

```
77    RESULTS-FLD  PIC 9(5)V99

77    SWITCH-FLD   PIC X VALUE 'N'.
```

2. Use the Level 66 RENAMES clause to rename the GROUP-CODE as GROUP-DESCRIPTION.

```
01    INVENTORY-RECORD.
10          PART-NO          PIC X(10).
10          GROUP-CODE.
            15   ID-1        PIC X(5).
            15   ID-2        PIC X(5).
            15   ID-3        PIC X(5).
10          DESC.            PIC X(25).
```

ADDITIONAL COBOL ELEMENTS

UNIT VIII

ASSIGNMENT SHEET 2: Name Qualification, BLANK WHEN ZERO, and Figurative Constants

1. Based upon the data presented in TM 2

   a. Move name-employee from the transaction-record to the master-record.

   b. Move corresponding transaction-record to master record. What field will not be moved?

2. Write a correct BLANK WHEN ZERO clause for the following report field.

   05 INVENTORY-AMT-ON-HAND PIC 9(5).

3. Write a statement to use the figurative constant SPACE to set a field called OUTPUT-LINE to blanks.

4. Write a statement to use the figurative constant HIGH-VALUES to set a field called TRAILER-ID to sort highest.

ADDITIONAL COBOL ELEMENTS

UNIT VIII

ASSIGNMENT SHEET 3: GO TO, ACCEPT/DISPLAY, and ON SIZE ERROR Statements

1. Write a GO TO DEPENDING ON statement for a field called GRADE. IF grade is one, GO TO paragraph-A; if grade is two, GO TO paragraph-B, etc.

2. Write a statement to ACCEPT the data and to DISPLAY it. Name of the field is DATE-FIELD.

3. Add the ON SIZE ERROR to the following divide statement:

```
DIVIDE QTY-ON-HAND BY BASE GIVING
         PERCENT-ON-HAND.
```

ADDITIONAL COBOL ELEMENTS

UNIT VIII

ANSWERS TO ASSIGNMENTS

Assignment Sheet 1

```
1.  01   RESULTS-FLD PIC 9(5)V99.
    01   SWITCH-FLD PIC 9 VALUE 'N'.
2.  66   GROUP-DESCRIPTION RENAMES GROUP-CODE.
```

Assignment Sheet 2

1. a. MOVE NAME-EMPLOYEE OF TRANSACTION-RECORD TO NAME-EMPLOYEE OF MASTER-RECORD.
   b. the field in transaction-record which does not have a corresponding filed in master-record
2. 05 INVENTORY-AMT-ON-HAND- PIC 9 (5) BLANK WHEN ZERO.
3. MOVE SPACE TO OUTPUT-LINE.
4. MOVE HIGH-VALUES TO TRAILER-ID.

Assignment Sheet 3

```
1.  GO TO
         PARAGRAPH-A
         PARAGRAPH-B
         PARAGRAPH-C
         PARAGRAPH-D
         PARAGRAPH-F
              DEPENDING ON GRADE.
2.  ACCEPT DATE-FIELD
    DISPLAY DATE-FIELD.
```

3. DIVIDE QTY-ON-HAND BY BASE GIVING PERCENT-ON-HAND ON SIZE ERROR any imperative statement.

ADDITIONAL COBOL ELEMENTS

UNIT VIII

EXAMINATION

1. Match each term in Column A with its correct phrase in Column B.

	Column A		Column B
____ a.	Level 77	1.	Gets low volume input
____ b.	Level 66	2.	Counts characters
____ c.	GO TO DEPENDING	3.	Prevents arithmetic overflow
____ d.	ON SIZE ERROR	4.	Case structure
____ e.	INSPECT	5.	Sorting order
____ f.	ACCEPT	6.	Independent data
____ g.	Collating sequence	7.	Renames data
		8.	Redefines data

2. Given the following incorrect Level 77:
   a. Correct it, and
   b. Recode it as an 01-level item.

```
77   TOTAL-YTD-GROSS-PAY PIC Z(5)V99 VALUE ZERO.
```

3. Your project leader wants you to update an old program. You are to code Level 66 entries. Below is a sample data field, which you are to code and rename as INVENTORY-VALUE:

```
05 I-VALUE PIC 9 (5) V999.
```

4. A nonunique field with the data-name of CUSTOMER-ID is to be moved to another field also called CUSTOMER-ID. Write a correctly qualified move sentence to accomplish this move. The first CUSTOMER-ID field is in a record called WS-MASTER; the second CUSTOMER-ID field is in a record called PRINT-MASTER.

5. Code a correct BLANK WHEN ZERO clause to the following data item which is in a record used in a report.

```
05 TOTAL-SPENT PIC $Z,ZZZ,ZZ9.99.
```

6. Write a statement to move the figurative constant HIGH-VALUES to a field called SORT-KEY.

7. Explain why the unconditional GO TO is to be avoided.

8. Explain why the ALTER statement is to be avoided.

9. Write a DISPLAY statement to display a field called TODAYS-DATE along with a message identifying it as today's date.

10. Write a correct ON SIZE ERROR for the following arithmetic statement which performs a routine called arithmetic-error.

    ADD TOTAL-GROSS TO YTD-GROSS.

11. Explain how an INSPECT statement is used in error control.

12. Explain the use of the STRING statement.

ADDITIONAL COBOL ELEMENTS

UNIT VIII

ANSWERS TO EXAMINATION

1.
   a. 6
   b. 7
   c. 4
   d. 3
   e. 2
   f. 1
   g. 5

2.
   a. 77 TOTAL-YTD-GROSS-PAY PIC S9(5)V99 VALUE ZERO.
   b. 01 TOTAL-YTD-GROSS-PAY PIC S9(5)V99 VALUE ZERO.

3. 66 INVENTORY-VALUE RENAMES I-VALUE.

4. MOVE CUSTOMER-ID OF WS-MASTER TO CUSTOMER-ID OF PRINT-MASTER.

5. 05 TOTAL-SPENT PIC $Z,ZZZ,ZZ9.99 BLANK WHEN ZERO.

6. MOVE HIGH-VALUES TO SORT-KEY.

7. It leads to fragmented, difficult to follow, and unmaintainable program structures.

8. Same as 7.

9. DISPLAY 'THE DATE TODAY IS' TODAY'S-DATE.

10. ADD TOTAL-GROSS TO YTD-GROSS ON SIZE ERROR PERFORM ARITHMETIC-ERROR.

11. INSPECT statement is used to examine input fields character-by-character looking for key stroke type errors.

12. String is used to put date fields together into grouped data.

SEQUENTIAL FILES I

UNIT IX

UNIT OBJECTIVE

After studying this unit, a student should be able to write edit routines and other portions of programs to create a sequential master file.

SPECIFIC OBJECTIVES

After studying this unit, a student should be able to:

1. Match terms and definitions.
2. Recognize correct statements about the edit checks in the sample edit program.
3. Explain the types of COBOL edit checks for input fields.
4. Recognize correct statements on the detailed programming specifications concerning the IPO and hierarchy charts for the sample program.
5. Recognize correct statements on the detailed programming specifications concerning the pseudocode for the sample program.
6. Write COBOL code similar to the program code for the sample.
7. Apply the preceding objectives in assignment sheets and job sheets.

SEQUENTIAL FILES I

UNIT IX

SUGGESTED ACTIVITIES

I. Instructor

   A. Make transparencies.

   B. Provide students with objectives and discuss.

   C. Provide students with information sheets, assignment sheets, and text assignments and discuss.

   D. Give an examination.

II. Student

   A. Read objectives.

   B. Study information sheets and text as assigned by instructor.

   C. Complete assignment sheets and program assignments.

   D. Complete examination.

INSTRUCTIONAL MATERIALS

I. Unit content

   A. Objectives

   B. Notes to the instructor

   C. Information sheets

   D. Handouts

      1. HO 1: Program Specifications (Parts 1, 2, 3, and 4)

      2. HO 2: Pseudocode for Edit Estimated Hours Module

      3. HO 3: Source Listing of Sample Program (Parts 1, 2, 3, and 4)

   E. Transparency masters

      1. TM 1: Creating a Sequential Master File

      2. TM 2: Application Program Master File Information

3. TM 3: Example of Check Totals (Parts 1 and 2)

4. TM 4: IPO Chart for Top-level Module

5. TM 5: Overall Hierarchy Chart

6. TM 6: Definition of Sequential File on Disk

7. TM 7: Example of the Use of Subscripted Fields (Parts 1, 2, and 3)

8. TM 8: Transaction File Input Record

9. TM 9: Example of Checking for Missing Data

F. Assignment sheets

1. AS 1: Edit Checks

2. AS 2: IPO Charts Summary

G. Answers to assignment sheets

H. Examination

I. Answers to examination

II. References

A. Shelly, Gary B. and Thomas J. Cashman. "Sequential File Processing; Data Editing" in Advanced Structured COBOL: Program Design and File Processing. Brea, CA: Anaheim Publishing Company, 1981.

SEQUENTIAL FILES I

UNIT IX

NOTES TO THE INSTRUCTOR

1. Detailed design specifications and development need further input from the instructor. If using the Shelly and Cashman, Advanced text, use pp. 2.20-2.34.

2. Programming assignments from the text would be appropriate in this unit.

3. You may prefer to have students use LABEL RECORDS OMITTED as this will simplify the JCL.

SEQUENTIAL FILES I

UNIT IX

INFORMATION SHEETS

I. Terms and functional definitions

   A. **Sequential file organization** - file which is accessed starting at the beginning and proceeding to the end in order of physical occurrence

   B. **Editing** - checking data for errors or unreasonable conditions to insure that only valid data is processed

   C. **Cross footing** - cross totaling a file or report and checking to see that the cross (independently derived) totals match

   D. **Reasonableness check** - edit check to determine if an input value or calculated result surpasses some preset terminal limit; similar to limits check

   E. **Check for numeric** - COBOL test for digits in a field only

II. Introduction to sequential file editing and sample editing problem

   A. Introduction

      1. Programs often fail because of bad input data. The purpose of editing is to detect and prevent the input of bad or erroneous data.

      2. When editing forms:

         a. Check for blank fields.

         b. Check for numeric in a field only.

         c. Check for specific values.

         d. Check cross total (cross footing).

         e. Check range.

   B. Sample editing problem

      1. The problem application is to edit input and create a master consisting of data about previously written program. (TM 1)

      2. Contents of master file of application programs (TM 2)

      3. The master file is to be on disk (TM 2). This same master file will be used in the next two units.

4. Input specifications for the problem are in TM 8.

5. Processing is defined in the programming specification HO 1. Note particularly the edit specifications.

6. Error exception report - transactions which do not meet the edit specifications are to be printed in this report.

III. Types of edits for input fields

A. Blank and Missing Data Check

1. Pseudocode form for this check is:

```
IF field-to-be checked is equal to spaces then do error-procedures.
```

2. Error-procedures will include:

   a. Setup ID of transaction in output

   b. Move name-of-the-field to output

   c. Move description of error to output

   d. Write report line

   e. Set error switch

3. TM 9 is an example of COBOL coding of a check and error-procedure.

B. Check for numeric data

1. The COBOL test for numeric not only checks that the data is there, (i.e., not blank) but is all digits.

2. Three types of checks for numeric

   a. On alphanumeric data (PIC X), a sign character cannot exist - i.e., must be all digits. (Note: This check is not allowed on some compilers. Data must be redefined as numeric; then check.)

   b. On numeric data (PIC 9), same as alphanumeric; must be all digits with no sign character allowed

   c. On signed numeric data (PIC S9), must be all digits but low-order (right most byte - may contain a sign character in addition to a digit)

C. Checks for specific values

1. Test that a specific value exists in a field.

2. Example: grade field must contain A, B, C, D, or F.

3. May be combined with Level 88's to give names to each of the possible values; e.g., GRADE-IS-A.

D. Check totals - cross footing

1. In the sample problem, the sum of the estimated hours for review, design, code, test and implementation must equal the total estimated hours field.

2. Also applied to totaling whole reports and files against each other for cross check

3. Study TM 3 for sample checks. Note especially the PERFORM VARYING at line 023070. If you are unfamiliar with this form, see your text on the topic of tables.

E. Reasonableness checks

1. Generally set by management; not an absolute rule but suggested limit

2. For example: hours worked in a single day greater than 22 may not be impossible but might be unreasonable depending upon management's rules.

IV. Design of IPO and hierarchy charts for sample program

A. TM 4 gives the overall or summary specifications for the application.

B. HO 1 gives the detailed specifications for the application.

C. IPO charts for each module were drawn to determine the details from the overall specifications. Your text will show further details or your instructor will provide assistance in going through this design process .

V. Design of pseudocode for sample program

A. Most edit modules are quite similar. This part is limited to one module - Edit Estimated Hours, D010 in TM 5. (See HO 2 for pseudocode)

B. In HO 2, notice the use of subscripts in the first IF. As there are six types of estimated hours, they are treated as if they were a one-dimensional table. Example: when subscript = 1, it refers to the first field, 2 refers to the second, etc.

C. Subscripts are also used to add each of the estimated hours in order to arrive at total estimated hours.

VI. Program coding and listing

A. Definition of sequential disk file

1. TM 6 gives a sample of the coding for defining a sequential disk file.

2. There are no special commands for sequential because in the absence of other options, sequential is assumed.

3. Note at line 004080 that this command calls for standard label records. Since this causes more complexities in your JCL, your instructor may prefer that you use LABEL RECORDS OMITTED.

4. In the 01, notice that the three groups of level 10's are examples of one-dimensional tables.

B. Table usage (TM 7)

C. The source listing for the sample program is given in HO 3.

D. Notes on listing:

1. Lines 7020-7090 are good examples of use of Level 88 as names for values of switch settings.

2. Observe the program constants starting in line 8050.

3. In the procedure division, notice the use of annotation for each major module. Also, the annotation tells the source of control for each module.

4. Study the edit modules carefully as these contain useful but complex logic.

311

HO 1

Program Specifications (Part 1 of 4)

Programming Specifications

SUBJECT Load Application Program Master File	**DATE** May 2	**PAGE 1 OF 4**
TO Programmer	**FROM** Systems Analyst	

A program is to be written to load the Application Program Master File. The format of the input record, the master file record, and the exception report are included as a part of this narrative. The program should be written to include the following processing.

1. The program should read the transaction input file and create the Application Program Master File as illustrated in the description of the master file record.
2. The Actual Review Hours, Actual Design Hours, Actual Code Hours, Actual Test Hours, and Actual Implementation Hours should each contain zeros.
3. The Review Status Code, Design Status Code, Code Status Code, Test Status Code, and Implementation Status Code fields should each contain a blank.
4. Before the information in the transaction input record is used to build the Application Program Master File record, the data in the record must be edited. If it passes the following editing criteria, then the data in the transaction input record should be used to create the master record. If any of the following tests is not met by the data in the transaction input record, then the entire transaction record must be rejected and not used to create a master record.

Program Specifications (Part 2 of 4)

Programming Specifications

SUBJECT Load Application Program Master File	**DATE** May 2	**PAGE 2 OF 4**
TO Programmer	**FROM** Systems Analyst	

- a. Program Identification
  1. The Application Designation must be one of the valid codes: AP (Accounts Payable), AR (Accounts Receivable), CA (Cost Accounting), GL (General Ledger), IC (Inventory Control), PA (Payroll). It should be noted that these application areas may expand, so provision should be made in the program for easily adding new application designation codes.
  2. The Program Identification Number must be numeric.
- b. Program Description
  1. The Program Description must be present. If the field is all blanks, it is in error.
- c. Programmer Employee Number
  1. The Programmer Employee number must be numeric.
- d. Total Estimated Hours
  1. The Total Estimated Hours must be numeric.
- e. Estimated Hours (Review, Design, Code, Test, Implementation)
  1. Each field must contain numeric data.
  2. The sum of the Estimated Hours must equal the value in the Total Estimated Hours field.

5. The Estimated Hours fields must also be checked to determine if they contain reasonable values. If it is found that they do not contain reasonable values, an entry should be made on the Exception Report; but the data in the transaction input record should be used to create the master record. The reasonable values for the hours are calculated as follows:

- a. Estimated Review Hours
  1. Maximum Reasonable Hours = Total Estimated Hours x 20%
  2. Minimum Reasonable Hours = Total Estimated Hours x 5%
- b. Estimated Design Hours
  1. Maximum Reasonable Hours = Total Estimated Hours x 60%
  2. Minimum Reasonable Hours = Total Estimated Hours x 30%

Program Specifications (Part 3 of 4)

Programming Specifications

SUBJECT Load Application Program Master File	**DATE** May 2	**PAGE 3 OF 4**
TO Programmer	**FROM** Systems Analyst	

c. Estimated Code Hours
   1. Maximum Reasonable Hours = Total Estimated Hours x 50%
   2. Minimum Reasonable Hours = Total Estimated Hours x 10%

d. Estimated Test Hours
   1. Maximum Reasonable Hours = Total Estimated Hours x 30%
   2. Minimum Reasonable Hours = Total Estimated Hours x 5%

e. Estimated Implementation Hours
   1. Maximum Reasonable Hours = Total Estimated Hours x 20%
   2. Minimum Reasonable Hours = Total Estimated Hours x 0%

If any of the Estimated Hours fields contains a value greater than the maximum reasonable hours or less than the minimum reasonable hours, then the appropriate message should be printed on the Exception Report, but the record should still be loaded onto the Application Program Master File.

6. All fields in each input record should be checked.

7. If any field is found to be in error, the appropriate entry should be made on the Exception Report. The fields on the Exception Report are explained below.
   a. Program I.D. - This field should contain the Program Identification of any record which contains one or more errors. It is to be group indicated on the report; that is, if a single input record contains more than one error, the Program Identification should appear for the first error on the first line only.
   b. Field in Error - This field should identify the field which contains the error. If the error "Sum of Estimated Hours Not = Total Hours" is found, the "Total Estimated Hours" field should be printed as the Field in Error.
   c. Type of Error - This field indicates the type of error which has been found in the input record.

Program Specifications (Part 4 of 4)

Programming Specifications

SUBJECT Load Application Program Master File	**DATE** May 2	**PAGE 4 OF 4**
TO Programmer	**FROM** Systems Analyst	

   d. The maximum number of lines on a single page of the Exception Report is forty (40) lines.

8. The program should be written in COBOL.

Pseudocode for Edit Estimated Hours Module

Pseudocode Specifications

DATE: May 9	PROGRAMMER: Jane Austin	PROGRAM: Load Program Master File
MODULE FUNCTION: Edit the Estimated Hours	REF: D010	MODULE NAME: Edit Estimated Hours

PSEUDOCODE / **REF:**

```
Set numeric indicator to indicate numeric fields
Set loop count to 1 (set subscript)
PERFORM UNTIL all individual estimated hours
        have been checked (count = 6)
  IF individual estimated hours (subscript) not numeric
    Move program identification to work area
    Move name of field indication (subscript) to work area
    Move 'non-numeric estimated hours' to work area
    Write exception report                                  U000
    Indicate non-numeric individual estimated hours
    Indicate record is not valid
  ENDIF
  Add 1 to count for hours to be checked
ENDPERFORM
IF total estimated hours are not numeric
  Move program identification to work area
  Move 'total estimated hours' to work area
  Move 'non-numeric total estimated hours' to work area
  Write exception report                                    U000
  Indicate record is not valid
ELSE
  Check individual estimated hours reasonableness           E000
  IF individual estimated hours are numeric
    Set work area for sum to zero
    Set count (i.e., subscript) to 1
    PERFORM UNTIL all fields have been added (count = 6)
      Add individual estimated hours (subscript) to work
        area
      Add 1 to count
    ENDPERFORM
    IF sum of individual estimated hours not = total
        estimated hours
      Move program identification to work area
      Move 'total estimated hours' to work area
      Move 'sum of estimated hours not = total
        hours' to work area
      Write exception report                                U000
      Indicate record is not valid
    ENDIF
  ENDIF
ENDIF
```

FILES, RECORDS, FIELDS REQUIRED

Numeric field indicator
Loop count (subscript) to reference estimated hours
Estimated hours (as table values)
Program identification
Program I.D. work area
'Review hours' constant
Field in error work area
'Design hours' constant
'Code hours' constant
'Test hours' constant
'Implementation hours' constant
'Non-numeric estimated hours' constant
Type of error work area
Valid record indicator
Total estimated hours
'Total estimated hours' constant
'Non-numeric total estimated hours' constant
Subscript (count to add all individual hours
'Sum of estimated hours not = total hours' constant

BEST COPY AVAILABLE

HO 3

Source Listing of Sample Program (Part 1)

```
                    8.45.34         JUN 16,1978

001010 IDENTIFICATION DIVISION.
001020
001030 PROGRAM-ID.    LOADMSTR.
001040 AUTHOR.        SHELLY AND CASHMAN.
001050 INSTALLATION.  ANAHEIM.
001060 DATE-WRITTEN.  05/10/78.
001070 DATE-COMPILED. JUN 16,1978.
001080 SECURITY.      UNCLASSIFIED.
001090
001100*****************************************************************
001110*                                                                *
001120*  THIS PROGRAM LOADS THE APPLICATION PROGRAM MASTER FILE. IT   *
001130*  READS THE TRANSACTION FILE AND EDITS THE FIELDS WITHIN THE   *
001140*  RECORD TO ENSURE THAT THEY CONTAIN VALID DATA. IF ALL THE    *
001150*  FIELDS CONTAIN VALID DATA, THE RECORD IS WRITTEN ON THE      *
001160*  APPLICATION PROGRAM MASTER FILE. IF ANY ERRORS (EXCEPT       *
001170*  REASONABLENESS CHECKS ON TASK HOURS) ARE FOUND, THE RECORD   *
001180*  IS NOT WRITTEN ON THE MASTER FILE. WHEN AN ERROR OR          *
001190*  VIOLATION OF THE REASONABLENESS CHECK OCCURS, A LINE IS      *
001200*  WRITTEN ON THE APPLICATION PROGRAM MASTER LOAD EXCEPTION     *
002010*  REPORT. ALL FIELDS ARE CHECKED IN EACH TRANSACTION.          *
002020*                                                                *
002030*****************************************************************
002040
002050
002060
002070 ENVIRONMENT DIVISION.
002080
002090 CONFIGURATION SECTION.
002100
002110 SOURCE-COMPUTER. IBM-370.
002120 OBJECT-COMPUTER. IBM-370.
002130 SPECIAL-NAMES.   C01 IS TO-THE-TOP-OF-THE-PAGE.
002140
002150 INPUT-OUTPUT SECTION.
002160
002170 FILE-CONTROL.
002180     SELECT PROGRAM-TRANSACTION-INPUT-FILE
002190         ASSIGN TO UR-S-SYSIN.
002200     SELECT PROGRAM-MASTER-OUTPUT-FILE
003010         ASSIGN TO DA-S-PGMMSTR.
003020     SELECT EXCEPTION-REPORT-FILE
003030         ASSIGN TO UR-S-SYSOUT.
003050 DATA DIVISION.
003060
003070 FILE SECTION.
003080
003090 FD  PROGRAM-TRANSACTION-INPUT-FILE
003100        RECORD CONTAINS 80 CHARACTERS
003110        LABEL RECORDS ARE OMITTED
003120        DATA RECORD IS PROGRAM-TRANSACTION-INPUT-REC.
003130
003140 01  PROGRAM-TRANSACTION-INPUT-REC.
003150     05  PROGRAM-IDENTIFICATION-TRANSIN.
003160         10  APPL-DESIGNATION-TRANSIN           PIC XX.
003170             88  APPL-DESIGNATION-IS-VALID       VALUE 'AP', 'AR',
003180                                                       'GL', 'PA',
003190                                                       'IC', 'CA'.
003200         10  IDENTIFICATION-NUMBER-TRANSIN      PIC X(4).
004010     05  PROGRAM-DESCRIPTION-TRANSIN            PIC X(20).
004020     05  PROGRAMMER-EMPL-NUMBER-TRANSIN         PIC X(6).
004030     05  TOTAL-ESTIMATED-HOURS-TRANSIN          PIC 9(4).
004040     05  ESTIMATED-TASK-HOURS-TRANSIN.
004050         10  ESTIMATED-REVIEW-HOURS-TRANSIN     PIC 9(4).
004060         10  ESTIMATED-DESIGN-HOURS-TRANSIN     PIC 9(4).
004070         10  ESTIMATED-CODE-HOURS-TRANSIN       PIC 9(4).
004080         10  ESTIMATED-TEST-HOURS-TRANSIN       PIC 9(4).
004090         10  ESTIMATED-IMPLEM-HOURS-TRANSIN     PIC 9(4).
004100     05  ESTIMATED-HOURS-TABLE-TRANSIN REDEFINES
004110           ESTIMATED-TASK-HOURS-TRANSIN         PIC 9(4)
004120                                                 OCCURS 5 TIMES.
004130     05  FILLER                                 PIC X(24).
004140
004150 FD  PROGRAM-MASTER-OUTPUT-FILE
004160        BLOCK CONTAINS 20 RECORDS
004170        RECORD CONTAINS 77 CHARACTERS
004180        LABEL RECORDS ARE STANDARD
004190        DATA RECORD IS PROGRAM-MASTER-OUTPUT-RECORD.
004200
005010 01  PROGRAM-MASTER-OUTPUT-RECORD.
005020     05  PROGRAM-IDENTIFICATION-MSTROUT.
005030         10  APPL-DESIGNATION-MSTROUT           PIC XX.
005040         10  IDENTIFICATION-NUMBER-MSTROUT      PIC X(4).
005050     05  PROGRAM-DESCRIPTION-MSTROUT            PIC X(20).
005060     05  PROGRAMMER-EMPL-NUMBER-MSTROUT         PIC X(6).
005070     05  ESTIMATED-TASK-HOURS-MSTROUT.
005080         10  ESTIMATED-REVIEW-HOURS-MSTROUT     PIC 9(4).
005090         10  ESTIMATED-DESIGN-HOURS-MSTROUT     PIC 9(4).
005100         10  ESTIMATED-CODE-HOURS-MSTROUT       PIC 9(4).
005110         10  ESTIMATED-TEST-HOURS-MSTROUT       PIC 9(4).
005120         10  ESTIMATED-IMPLEM-HOURS-MSTROUT     PIC 9(4).
005130     05  ACTUAL-TASK-HOURS-MSTROUT.
005140         10  ACTUAL-REVIEW-HOURS-MSTROUT        PIC X(4).
005150         10  ACTUAL-DESIGN-HOURS-MSTROUT        PIC X(4).
005160         10  ACTUAL-CODE-HOURS-MSTROUT          PIC X(4).
005170         10  ACTUAL-TEST-HOURS-MSTROUT          PIC X(4).
005180         10  ACTUAL-IMPLEM-HOURS-MSTROUT        PIC X(4).
005190     05  TASK-STATUS-INDICATORS-MSTROUT.
005200         10  REVIEW-STATUS-MSTROUT              PIC X.
006010         10  DESIGN-STATUS-MSTROUT              PIC X.
006020         10  CODE-STATUS-MSTROUT                PIC X.
006030         10  TEST-STATUS-MSTROUT                PIC X.
006040         10  IMPLEM-STATUS-MSTROUT              PIC X.
006050
006060 FD  EXCEPTION-REPORT-FILE
006070        RECORD CONTAINS 133 CHARACTERS
006080        LABEL RECORDS ARE OMITTED
006090        DATA RECORD IS EXCEPTION-REPORT-LINE.
006100
006110 01  EXCEPTION-REPORT-LINE.
006120     05  CARRIAGE-CONTROL                          PIC X.
006130     05  PROGRAM-IDENTIFICATION-REPORT             PIC X(6).
006140     05  FILLER                                    PIC X(4).
006150     05  FIELD-IN-ERROR-REPORT                     PIC X(30).
006160     05  FILLER                                    PIC X(4).
006170     05  TYPE-OF-ERROR-REPORT                      PIC X(45).
006180     05  FILLER                                    PIC X(43).
006190
006200 WORKING-STORAGE SECTION.
007010
007020 01  PROGRAM-INDICATORS.
007030     05  ARE-THERE-MORE-RECORDS        PIC XXX  VALUE 'YES'.
007040         88  THERE-ARE-NO-MORE-RECORDS          VALUE 'NO '.
007050     05  IS-THE-RECORD-VALID           PIC XXX  VALUE SPACES.
007060         88  A-VALID-RECORD-IS-FOUND            VALUE 'YES'.
007070         88  THE-RECORD-IS-NOT-VALID            VALUE 'NO '.
007080     05  ARE-ESTIMATED-HOURS-NUMERIC PIC XXX  VALUE 'YES'.
007090         88  ESTIMATED-HOURS-ARE-NUMERIC        VALUE 'YES'.
007100
007110 01  EXCEPTION-REPORT-COMPARE-AREA.
007120     05  PREVIOUS-PROGRAM-ID           PIC X(6) VALUE LOW-VALUES.
007130
007140 01  PROGRAM-WORK-AREAS.
007150     05  PROGRAM-ID-WORK-AREA       PIC X(6).
007160     05  FIELD-IN-ERROR-WORK-AREA   PIC X(30).
007170     05  TYPE-OF-ERROR-WORK-AREA    PIC X(45).
007180     05  SUM-OF-ESTIMATED-HOURS-WORK PIC S9(5) VALUE ZERO
007190                                         USAGE IS COMP-3.
008040
008050 01  PROGRAM-CONSTANTS.
008060     05  DESCRIPTION-FIELD-MSG         PIC X(30) VALUE
008070                                   'DESCRIPTION
008080     05  DESCRIPTION-ERROR-MSG         PIC X(45) VALUE
008090                'NO DESCRIPTION PRESENT
008100     05  PROGRAMMER-EMPL-NUMBER-MSG  PIC X(30) VALUE
008110                                   'PROGRAMMER EMPLOYEE NUMBER
008120     05  PROGRAMMER-NUMBER-ERROR-MSG PIC X(45) VALUE
008130                'NON-NUMERIC PROGRAMMER EMPLOYEE NUMBER
008140     05  APPLICATION-DESIGNATION-MSG PIC X(30) VALUE
008150                                   'APPLICATION DESIGNATION
008160     05  APPL-DESIGNATION-ERROR-MSG  PIC X(45) VALUE
008170                'INVALID APPLICATION DESIGNATION
008180     05  IDENTIFICATION-NUMBER-MSG   PIC X(30) VALUE
008190                                   'PROGRAM IDENTIFICATION NUMBER
008200     05  IDENTIFICATION-NUMBER-ERR-MSG PIC X(45) VALUE
009010                'NON-NUMERIC IDENTIFICATION NUMBER
009020     05  ESTIMATED-HOURS-FIELD-MSGS
009030         10  REVIEW-HOURS-FIELD-MSG  PIC X(30) VALUE
009040                                   'REVIEW HOURS
009050         10  DESIGN-HOURS-FIELD-MSG  PIC X(30) VALUE
009060                                   'DESIGN HOURS
009070         10  CODE-HOURS-FIELD-MSG    PIC X(30) VALUE
009080                                   'CODE HOURS
009090         10  TEST-HOURS-FIELD-MSG    PIC X(30) VALUE
009100                                   'TEST HOURS
009110         10  IMPLEM-HOURS-FIELD-MSG  PIC X(30) VALUE
009120                                   'IMPLEMENTATION HOURS
009130     05  ALL-ESTIMATED-HOURS-FIELD-MSGS REDEFINES
009140           ESTIMATED-HOURS-FIELD-MSGS PIC X(30) OCCURS 5 TIMES.
009150     05  ESTIMATED-HOURS-ERROR-MSG   PIC X(45) VALUE
009160                'NON-NUMERIC ESTIMATED HOURS
009170     05  TOTAL-EST-HOURS-FIELD-MSG   PIC X(30) VALUE
009180                                   'TOTAL ESTIMATED HOURS
009190     05  TOTAL-EST-HOURS-ERROR-MSG   PIC X(45) VALUE
009200                'NON-NUMERIC TOTAL ESTIMATED HOURS
009210     05  ESTIMATED-HOURS-SUM-ERROR-MSG PIC X(45) VALUE
009220                'SUM OF ESTIMATED HOURS NOT = TOTAL HOURS
009230     05  MAXIMUM-PERCENT-FOR-REVIEW  PIC V99   VALUE .20
010010                                              USAGE IS COMP-3.
010020     05  MINIMUM-PERCENT-FOR-REVIEW  PIC V99   VALUE .05
010030                                              USAGE IS COMP-3.
010040     05  MAXIMUM-PERCENT-FOR-DESIGN  PIC V99   VALUE .40
010050                                              USAGE IS COMP-3.
010060     05  MINIMUM-PERCENT-FOR-DESIGN  PIC V99   VALUE .30
010070                                              USAGE IS COMP-3.
010080     05  MAXIMUM-PERCENT-FOR-CODE    PIC V99   VALUE .50
010090                                              USAGE IS COMP-3.
010100     05  MINIMUM-PERCENT-FOR-CODE    PIC V99   VALUE .10
010110                                              USAGE IS COMP-3.
010120     05  MAXIMUM-PERCENT-FOR-TEST    PIC V99   VALUE .30
010130                                              USAGE IS COMP-3.
010140     05  MINIMUM-PERCENT-FOR-TEST    PIC V99   VALUE .05
010150                                              USAGE IS COMP-3.
010160     05  MAXIMUM-PERCENT-FOR-IMPLEM  PIC V99   VALUE .20
010170                                              USAGE IS COMP-3.
010180     05  MINIMUM-PERCENT-FOR-IMPLEM  PIC V99   VALUE .00
010190                                              USAGE IS COMP-3.
010200     05  REASONABLENESS-CHECK-ERROR-MSG PIC X(45) VALUE
011010                'VIOLATES REASONABLENESS CHECK - WARNING ONLY '.
011020
011030 01  PROGRAM-SUBSCRIPTS.
011040     05  HOURS-NUMERIC-CHECK-SUBSCRIPT  PIC S9(8) USAGE IS COMP
011050                                                   SYNC.
011060         88  ALL-HOURS-HAVE-BEEN-CHECKED            VALUE +6.
011070     05  SUM-ESTIMATED-HOURS-SUBSCRIPT  PIC S9(8) USAGE IS COMP
011080                                                   SYNC.
011090         88  ALL-HOURS-HAVE-BEEN-ADDED              VALUE +6.
011100
011110 01  PRINTER-CONTROL.
```

HO 3 (cont.)

Source Listing of Sample Program (Part 2)

```
           05  PROPER-SPACING           PIC 9.
           05  SPACE-ONE-LINE           PIC 9         VALUE 1.
           05  SPACE-TWO-LINES          PIC 9         VALUE 2.
           05  NUMBER-OF-LINES-PRINTED  PIC S999      VALUE ZERO
                                                      USAGE IS COMP-3.
           05  PAGE-SIZE                PIC 999       VALUE 40
                                                      USAGE IS COMP-3.
           05  PAGE-NUMBER-COUNT        PIC S999      VALUE +1
                                                      USAGE IS COMP-3.
               88  THIS-IS-THE-FIRST-PAGE             VALUE +1.

       01  HEADING-LINES.
           05  FIRST-HEADING-LINE.
               10  CARRIAGE-CONTROL     PIC X.
               10  CURRENT-DATE-MOD1    PIC X(8).
               10  FILLER               PIC X(9)      VALUE SPACES.
               10  FILLER               PIC X(12)     VALUE
                                                      'APPLICATION '.
               10  FILLER               PIC X(8)      VALUE 'PROGRAM '.
               10  FILLER               PIC X(7)      VALUE 'MASTER '.
               10  FILLER               PIC X(5)      VALUE 'LOAD '.
               10  FILLER               PIC X(10)     VALUE 'EXCEPTION '.
               10  FILLER               PIC X(6)      VALUE 'REPORT'.
               10  FILLER               PIC X(8)      VALUE SPACES.
               10  FILLER               PIC X(5)      VALUE 'PAGE '.
               10  PAGE-NUMBER-MOD1     PIC ZZ9.
               10  FILLER               PIC X(51)     VALUE SPACES.
           05  SECOND-HEADING-LINE.
               10  CARRIAGE-CONTROL     PIC X.
               10  FILLER               PIC X(7)      VALUE 'PROGRAM'.
               10  FILLER               PIC X(10)     VALUE SPACES.
               10  FILLER               PIC X(5)      VALUE 'FIELD'.
               10  FILLER               PIC X(37)     VALUE SPACES.
               10  FILLER               PIC X(4)      VALUE 'TYPE'.
               10  FILLER               PIC X(69)     VALUE SPACES.
           05  THIRD-HEADING-LINE.
               10  CARRIAGE-CONTROL     PIC X.
               10  FILLER               PIC X         VALUE SPACE.
               10  FILLER               PIC X(4)      VALUE 'I.D.'.
               10  FILLER               PIC X(10)     VALUE SPACES.
               10  FILLER               PIC X(8)      VALUE 'IN ERROR'.
               10  FILLER               PIC X(34)     VALUE SPACES.
               10  FILLER               PIC X(8)      VALUE 'OF ERROR'.
               10  FILLER               PIC X(67)     VALUE SPACES.
       PROCEDURE DIVISION.

      ****************************************************************
      *                                                              *
      *  THE FUNCTION OF THIS MODULE IS TO LOAD THE APPLICATION      *
      *  PROGRAM MASTER FILE. IT IS ENTERED FROM AND EXITS TO THE    *
      *  OPERATING SYSTEM.                                           *
      *                                                              *
      ****************************************************************

       A000-LOAD-PROGRAM-MASTER-FILE.

           OPEN INPUT  PROGRAM-TRANSACTION-INPUT-FILE
                OUTPUT PROGRAM-MASTER-OUTPUT-FILE
                       EXCEPTION-REPORT-FILE.
           PERFORM B000-OBTAIN-VALID-INPUT-RECORD.
           PERFORM A001-WRITE-AND-READ
               UNTIL THERE-ARE-NO-MORE-RECORDS.
           CLOSE PROGRAM-TRANSACTION-INPUT-FILE
                 PROGRAM-MASTER-OUTPUT-FILE
                 EXCEPTION-REPORT-FILE.
           STOP RUN.



       A001-WRITE-AND-READ.

           PERFORM B010-WRITE-MASTER-FILE.
           PERFORM B000-OBTAIN-VALID-INPUT-RECORD.
      ****************************************************************
      *                                                              *
      *  THE FUNCTION OF THIS MODULE IS TO OBTAIN A VALID PROGRAM    *
      *  TRANSACTION INPUT RECORD. IT IS ENTERED FROM AND EXITS TO   *
      *  THE A000-LOAD-PROGRAM-MASTER-FILE MODULE.                   *
      *                                                              *
      ****************************************************************

       B000-OBTAIN-VALID-INPUT-RECORD.

           MOVE SPACES TO IS-THE-RECORD-VALID.
           PERFORM B001-READ-A-TRANSACTION-RECORD.
           PERFORM B002-EDIT-AND-READ
               UNTIL A-VALID-RECORD-IS-FOUND OR
                     THERE-ARE-NO-MORE-RECORDS.



       B001-READ-A-TRANSACTION-RECORD.

           READ PROGRAM-TRANSACTION-INPUT-FILE
               AT END
                   MOVE 'NO ' TO ARE-THERE-MORE-RECORDS.



       B002-EDIT-AND-READ.

           PERFORM C000-EDIT-INPUT-RECORD.
           IF THE-RECORD-IS-NOT-VALID
               MOVE SPACES TO IS-THE-RECORD-VALID
               PERFORM B001-READ-A-TRANSACTION-RECORD
           ELSE
               MOVE 'YES' TO IS-THE-RECORD-VALID.
```

```
      ****************************************************************
      *                                                              *
      *  THE FUNCTION OF THIS MODULE IS TO WRITE THE APPLICATION     *
      *  PROGRAM MASTER FILE. IT IS ENTERED FROM AND EXITS TO THE    *
      *  A000-LOAD-PROGRAM-MASTER-FILE MODULE.                       *
      *                                                              *
      ****************************************************************

       B010-WRITE-MASTER-FILE.

           MOVE PROGRAM-IDENTIFICATION-TRANSIN TO
                PROGRAM-IDENTIFICATION-MSTROUT.
           MOVE PROGRAM-DESCRIPTION-TRANSIN TO
                PROGRAM-DESCRIPTION-MSTROUT.
           MOVE PROGRAMMER-EMPL-NUMBER-TRANSIN TO
                PROGRAMMER-EMPL-NUMBER-MSTROUT.
           MOVE ESTIMATED-REVIEW-HOURS-TRANSIN TO
                ESTIMATED-REVIEW-HOURS-MSTROUT.
           MOVE ESTIMATED-DESIGN-HOURS-TRANSIN TO
                ESTIMATED-DESIGN-HOURS-MSTROUT.
           MOVE ESTIMATED-CODE-HOURS-TRANSIN TO
                ESTIMATED-CODE-HOURS-MSTROUT.
           MOVE ESTIMATED-TEST-HOURS-TRANSIN TO
                ESTIMATED-TEST-HOURS-MSTROUT.
           MOVE ESTIMATED-IMPLEM-HOURS-TRANSIN TO
                ESTIMATED-IMPLEM-HOURS-MSTROUT.
           MOVE ZERO TO ACTUAL-REVIEW-HOURS-MSTROUT
                        ACTUAL-DESIGN-HOURS-MSTROUT
                        ACTUAL-CODE-HOURS-MSTROUT
                        ACTUAL-TEST-HOURS-MSTROUT
                        ACTUAL-IMPLEM-HOURS-MSTROUT.
           MOVE SPACES TO REVIEW-STATUS-MSTROUT
                          DESIGN-STATUS-MSTROUT
                          CODE-STATUS-MSTROUT
                          TEST-STATUS-MSTROUT
                          IMPLEM-STATUS-MSTROUT.
           WRITE PROGRAM-MASTER-OUTPUT-RECORD.
      ****************************************************************
      *                                                              *
      *  THIS MODULE EDITS THE TRANSACTION INPUT RECORD. IT IS       *
      *  ENTERED FROM AND EXITS TO THE B000-OBTAIN-VALID-INPUT-RECORD*
      *  MODULE.                                                     *
      *                                                              *
      ****************************************************************

       C000-EDIT-INPUT-RECORD.

           PERFORM D000-EDIT-PROGRAM-ID.
           IF PROGRAM-DESCRIPTION-TRANSIN = SPACES
               MOVE PROGRAM-IDENTIFICATION-TRANSIN TO
                    PROGRAM-ID-WORK-AREA
               MOVE DESCRIPTION-FIELD-MSG TO FIELD-IN-ERROR-WORK-AREA
               MOVE DESCRIPTION-ERROR-MSG TO TYPE-OF-ERROR-WORK-AREA
               PERFORM U000-WRITE-EXCEPTION-REPORT
               MOVE 'NO ' TO IS-THE-RECORD-VALID.
           IF PROGRAMMER-EMPL-NUMBER-TRANSIN IS NOT NUMERIC
               MOVE PROGRAM-IDENTIFICATION-TRANSIN TO
                    PROGRAM-ID-WORK-AREA
               MOVE PROGRAMMER-EMPL-NUMBER-MSG TO
                    FIELD-IN-ERROR-WORK-AREA
               MOVE PROGRAMMER-NUMBER-ERROR-MSG TO
                    TYPE-OF-ERROR-WORK-AREA
               PERFORM U000-WRITE-EXCEPTION-REPORT
               MOVE 'NO ' TO IS-THE-RECORD-VALID.
           PERFORM D010-EDIT-ESTIMATED-HOURS.
      ****************************************************************
      *                                                              *
      *  THE FUNCTION OF THIS MODULE IS TO EDIT THE PROGRAM          *
      *  IDENTIFICATION. IT IS ENTERED FROM AND EXITS TO THE         *
      *  C000-EDIT-INPUT-RECORD.                                     *
      *                                                              *
      ****************************************************************

       D000-EDIT-PROGRAM-ID.

           IF APPL-DESIGNATION-IS-VALID
               NEXT SENTENCE
           ELSE
               MOVE PROGRAM-IDENTIFICATION-TRANSIN TO
                    PROGRAM-ID-WORK-AREA
               MOVE APPLICATION-DESIGNATION-MSG TO
                    FIELD-IN-ERROR-WORK-AREA
               MOVE APPL-DESIGNATION-ERROR-MSG TO
                    TYPE-OF-ERROR-WORK-AREA
               PERFORM U000-WRITE-EXCEPTION-REPORT
               MOVE 'NO ' TO IS-THE-RECORD-VALID.
           IF IDENTIFICATION-NUMBER-TRANSIN IS NOT NUMERIC
               MOVE PROGRAM-IDENTIFICATION-TRANSIN TO
                    PROGRAM-ID-WORK-AREA
               MOVE IDENTIFICATION-NUMBER-MSG TO
                    FIELD-IN-ERROR-WORK-AREA
               MOVE IDENTIFICATION-NUMBER-ERR-MSG TO
                    TYPE-OF-ERROR-WORK-AREA
               PERFORM U000-WRITE-EXCEPTION-REPORT
               MOVE 'NO ' TO IS-THE-RECORD-VALID.
      ****************************************************************
      *                                                              *
      *  THIS MODULE EDITS THE ESTIMATED HOURS IN THE INPUT RECORD.  *
      *  IT IS ENTERED FROM AND EXITS TO THE C000-EDIT-INPUT-RECORD  *
      *  MODULE.                                                     *
      *                                                              *
      ****************************************************************

       D010-EDIT-ESTIMATED-HOURS.

           MOVE 'YES' TO ARE-ESTIMATED-HOURS-NUMERIC.
           PERFORM D011-CHECK-FOR-NUMERIC-HOURS
```

BEST COPY AVAILABLE

Source Listing of Sample Program (Part 3)

```
           VARYING HOURS-NUMERIC-CHECK-SUBSCRIPT FROM 1 BY 1
           UNTIL ALL-HOURS-HAVE-BEEN-CHECKED.
     IF TOTAL-ESTIMATED-HOURS-TRANSIN IS NOT NUMERIC
        MOVE PROGRAM-IDENTIFICATION-TRANSIN TO
              PROGRAM-ID-WORK-AREA
        MOVE TOTAL-EST-HOURS-FIELD-MSG TO
              FIELD-IN-ERROR-WORK-AREA
        MOVE TOTAL-EST-HOURS-ERROR-MSG TO
              TYPE-OF-ERROR-WORK-AREA
        PERFORM U000-WRITE-EXCEPTION-REPORT
        MOVE 'NO ' TO IS-THE-RECORD-VALID
     ELSE
        PERFORM E000-CHECK-HOUR-REASONABLENESS
        IF ESTIMATED-HOURS-ARE-NUMERIC
           MOVE ZEROS TO SUM-OF-ESTIMATED-HOURS-WORK
           PERFORM D012-ADD-THE-HOURS
                 VARYING SUM-ESTIMATED-HOURS-SUBSCRIPT FROM 1 BY
                 UNTIL ALL-HOURS-HAVE-BEEN-ADDED
           IF SUM-OF-ESTIMATED-HOURS-WORK IS NOT =
                 TOTAL-ESTIMATED-HOURS-TRANSIN
              MOVE PROGRAM-IDENTIFICATION-TRANSIN TO
                    PROGRAM-ID-WORK-AREA
              MOVE TOTAL-EST-HOURS-FIELD-MSG TO
                    FIELD-IN-ERROR-WORK-AREA
              MOVE ESTIMATED-HOURS-SUM-ERROR-MSG TO
                    TYPE-OF-ERROR-WORK-AREA
              PERFORM U000-WRITE-EXCEPTION-REPORT
              MOVE 'NO ' TO IS-THE-RECORD-VALID.



 D011-CHECK-FOR-NUMERIC-HOURS.

     IF ESTIMATED-HOURS-TABLE-TRANSIN
           (HOURS-NUMERIC-CHECK-SUBSCRIPT) IS NOT NUMERIC
        MOVE PROGRAM-IDENTIFICATION-TRANSIN TO
              PROGRAM-ID-WORK-AREA
        MOVE ALL-ESTIMATED-HOURS-FIELD-MSGS
              (HOURS-NUMERIC-CHECK-SUBSCRIPT) TO
              FIELD-IN-ERROR-WORK-AREA
        MOVE ESTIMATED-HOURS-ERROR-MSG TO
              TYPE-OF-ERROR-WORK-AREA
        PERFORM U000-WRITE-EXCEPTION-REPORT
        MOVE 'NO ' TO ARE-ESTIMATED-HOURS-NUMERIC
        MOVE 'NO ' TO IS-THE-RECORD-VALID.



 D012-ADD-THE-HOURS.

     ADD ESTIMATED-HOURS-TABLE-TRANSIN
           (SUM-ESTIMATED-HOURS-SUBSCRIPT) TO
           SUM-OF-ESTIMATED-HOURS-WORK.
****************************************************************
*                                                              *
*  THIS MODULE CHECKS THE REASONABLENESS OF THE ESTIMATED HOURS*
*  IN THE INPUT RECORD. IT IS ENTERED FROM AND EXITS TO THE    *
*  D010-EDIT-ESTIMATED-HOURS MODULE.                           *
*                                                              *
****************************************************************

 E000-CHECK-HOUR-REASONABLENESS.

     PERFORM F000-CHECK-REVIEW-HOURS.
     PERFORM F010-CHECK-DESIGN-HOURS.
     PERFORM F020-CHECK-CODE-HOURS.
     PERFORM F030-CHECK-TEST-HOURS.
     PERFORM F040-CHECK-IMPLEM-HOURS.
****************************************************************
*                                                              *
*  THIS MODULE CHECKS THE REVIEW HOURS FOR REASONABLENESS. IT  *
*  IS ENTERED FROM AND EXITS TO THE                            *
*  E000-CHECK-HOUR-REASONABLENESS MODULE.                      *
*                                                              *
****************************************************************

 F000-CHECK-REVIEW-HOURS.

     IF ESTIMATED-REVIEW-HOURS-TRANSIN IS NUMERIC
        IF ESTIMATED-REVIEW-HOURS-TRANSIN IS GREATER THAN
              (TOTAL-ESTIMATED-HOURS-TRANSIN *
               MAXIMUM-PERCENT-FOR-REVIEW) OR
               IS LESS THAN (TOTAL-ESTIMATED-HOURS-TRANSIN *
              MINIMUM-PERCENT-FOR-REVIEW)
           MOVE PROGRAM-IDENTIFICATION-TRANSIN TO
                 PROGRAM-ID-WORK-AREA
           MOVE REVIEW-HOURS-FIELD-MSG TO
                 FIELD-IN-ERROR-WORK-AREA
           MOVE REASONABLENESS-CHECK-ERROR-MSG TO
                 TYPE-OF-ERROR-WORK-AREA
           PERFORM U000-WRITE-EXCEPTION-REPORT.
****************************************************************
*                                                              *
*  THIS MODULE CHECKS THE REASONABLENESS OF THE DESIGN HOURS.  *
*  IT IS ENTERED FROM AND EXITS TO THE                         *
*  E000-CHECK-HOUR-REASONABLENESS MODULE.                      *
*                                                              *
****************************************************************

 F010-CHECK-DESIGN-HOURS.

     IF ESTIMATED-DESIGN-HOURS-TRANSIN IS NUMERIC
        IF ESTIMATED-DESIGN-HOURS-TRANSIN IS GREATER THAN
              (TOTAL-ESTIMATED-HOURS-TRANSIN *
               MAXIMUM-PERCENT-FOR-DESIGN) OR
               IS LESS THAN (TOTAL-ESTIMATED-HOURS-TRANSIN *
              MINIMUM-PERCENT-FOR-DESIGN)
           MOVE PROGRAM-IDENTIFICATION-TRANSIN TO
                 PROGRAM-ID-WORK-AREA
           MOVE DESIGN-HOURS-FIELD-MSG TO
                 FIELD-IN-ERROR-WORK-AREA
           MOVE REASONABLENESS-CHECK-ERROR-MSG TO
                 TYPE-OF-ERROR-WORK-AREA
           PERFORM U000-WRITE-EXCEPTION-REPORT.
****************************************************************
*                                                              *
*  THIS MODULE CHECKS THE CODE HOURS FOR REASONABLE VALUES. IT *
*  IS ENTERED FROM AND EXITS TO THE                            *
*  E000-CHECK-HOUR-REASONABLENESS MODULE.                      *
*                                                              *
****************************************************************

 F020-CHECK-CODE-HOURS.

     IF ESTIMATED-CODE-HOURS-TRANSIN IS NUMERIC
        IF ESTIMATED-CODE-HOURS-TRANSIN IS GREATER THAN
              (TOTAL-ESTIMATED-HOURS-TRANSIN *
               MAXIMUM-PERCENT-FOR-CODE) OR
               IS LESS THAN (TOTAL-ESTIMATED-HOURS-TRANSIN *
              MINIMUM-PERCENT-FOR-CODE)
           MOVE PROGRAM-IDENTIFICATION-TRANSIN TO
                 PROGRAM-ID-WORK-AREA
           MOVE CODE-HOURS-FIELD-MSG TO
                 FIELD-IN-ERROR-WORK-AREA
           MOVE REASONABLENESS-CHECK-ERROR-MSG TO
                 TYPE-OF-ERROR-WORK-AREA
           PERFORM U000-WRITE-EXCEPTION-REPORT.
****************************************************************
*                                                              *
*  THIS MODULE CHECKS THE TEST HOURS FOR REASONABLENESS. IT IS *
*  ENTERED FROM AND EXITS TO THE E000-CHECK-HOUR-REASONABLENESS*
*  MODULE.                                                     *
*                                                              *
****************************************************************

 F030-CHECK-TEST-HOURS.

     IF ESTIMATED-TEST-HOURS-TRANSIN IS NUMERIC
        IF ESTIMATED-TEST-HOURS-TRANSIN IS GREATER THAN
              (TOTAL-ESTIMATED-HOURS-TRANSIN *
               MAXIMUM-PERCENT-FOR-TEST) OR
               IS LESS THAN (TOTAL-ESTIMATED-HOURS-TRANSIN *
              MINIMUM-PERCENT-FOR-TEST)
           MOVE PROGRAM-IDENTIFICATION-TRANSIN TO
                 PROGRAM-ID-WORK-AREA
           MOVE TEST-HOURS-FIELD-MSG TO
                 FIELD-IN-ERROR-WORK-AREA
           MOVE REASONABLENESS-CHECK-ERROR-MSG TO
                 TYPE-OF-ERROR-WORK-AREA
           PERFORM U000-WRITE-EXCEPTION-REPORT.
****************************************************************
*                                                              *
*  THIS MODULE CHECKS THE REASONABLENESS OF THE IMPLEMENTATION *
*  HOURS. IT IS ENTERED FROM AND EXITS TO THE                  *
*  E000-CHECK-HOUR-REASONABLENESS MODULE.                      *
*                                                              *
****************************************************************

 F040-CHECK-IMPLEM-HOURS.

     IF ESTIMATED-IMPLEM-HOURS-TRANSIN IS NUMERIC
        IF ESTIMATED-IMPLEM-HOURS-TRANSIN IS GREATER THAN
              (TOTAL-ESTIMATED-HOURS-TRANSIN *
               MAXIMUM-PERCENT-FOR-IMPLEM) OR
               IS LESS THAN (TOTAL-ESTIMATED-HOURS-TRANSIN *
              MINIMUM-PERCENT-FOR-IMPLEM)
           MOVE PROGRAM-IDENTIFICATION-TRANSIN TO
                 PROGRAM-ID-WORK-AREA
           MOVE IMPLEM-HOURS-FIELD-MSG TO
                 FIELD-IN-ERROR-WORK-AREA
           MOVE REASONABLENESS-CHECK-ERROR-MSG TO
                 TYPE-OF-ERROR-WORK-AREA
           PERFORM U000-WRITE-EXCEPTION-REPORT.
****************************************************************
*                                                              *
*  THIS MODULE IS ENTERED TO PRINT THE EXCEPTION REPORT. IT IS *
*  A UTILITY MODULE AND IS ENTERED FROM AND EXITS TO SEVERAL   *
*  MODULES WITHIN THE PROGRAM (SEE HIERARCHY CHART).           *
*                                                              *
****************************************************************

 U000-WRITE-EXCEPTION-REPORT.

     IF THIS-IS-THE-FIRST-PAGE OR
           NUMBER-OF-LINES-PRINTED IS = PAGE-SIZE OR
            IS GREATER THAN PAGE-SIZE
        PERFORM U010-PRINT-HEADINGS
        MOVE SPACES TO EXCEPTION-REPORT-LINE
        MOVE PROGRAM-ID-WORK-AREA TO
              PROGRAM-IDENTIFICATION-REPORT.
     IF PROGRAM-ID-WORK-AREA NOT = PREVIOUS-PROGRAM-ID
        MOVE PROGRAM-ID-WORK-AREA TO
              PROGRAM-IDENTIFICATION-REPORT
        MOVE PROGRAM-ID-WORK-AREA TO PREVIOUS-PROGRAM-ID
        MOVE SPACE-TWO-LINES TO PROPER-SPACING.
     MOVE FIELD-IN-ERROR-WORK-AREA TO FIELD-IN-ERROR-REPORT.
     MOVE TYPE-OF-ERROR-WORK-AREA TO TYPE-OF-ERROR-REPORT.
     WRITE EXCEPTION-REPORT-LINE
           AFTER PROPER-SPACING.
     ADD PROPER-SPACING TO NUMBER-OF-LINES-PRINTED.
     MOVE SPACE-ONE-LINE TO PROPER-SPACING.
     MOVE SPACES TO EXCEPTION-REPORT-LINE.
```

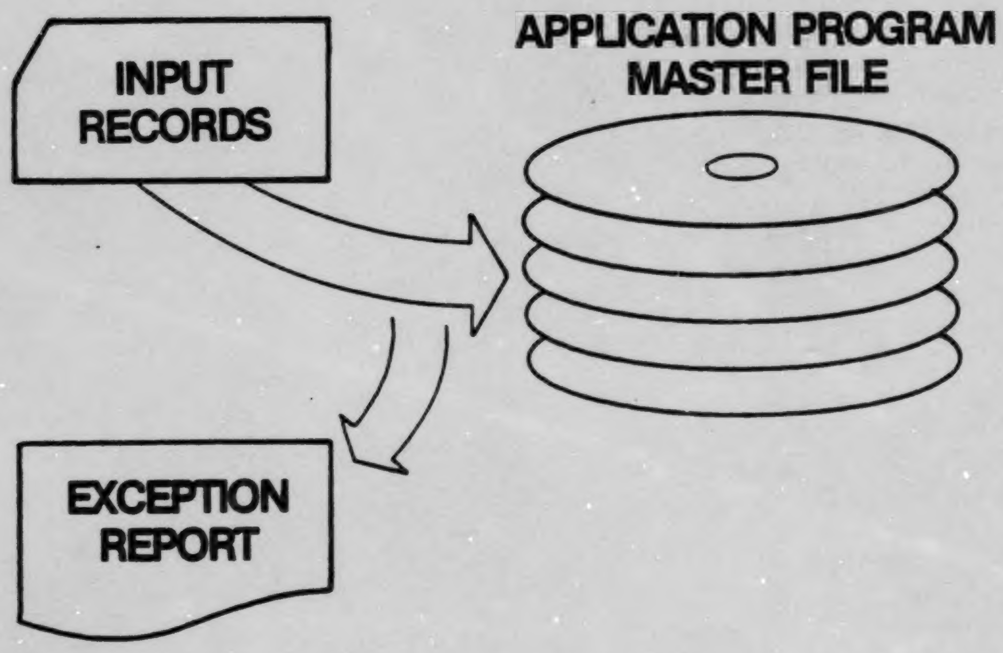
BEST COPY AVAILABLE

Source Listing of Sample Program (Part 4)

```
032190**************************************************************
032200*                                                            *
033010*  THIS MODULE PRINTS THE HEADINGS ON THE EXCEPTION REPORT. IT *
033020*  IS ENTERED FROM AND EXITS TO THE U000-WRITE-EXCEPTION-REPORT*
033030*  MODULE.                                                   *
033040*                                                            *
033050**************************************************************
033060
033070 U010-PRINT-HEADINGS.
033080
033090     MOVE CURRENT-DATE TO CURRENT-DATE-HDG1.
033100     MOVE PAGE-NUMBER-COUNT TO PAGE-NUMBER-HDG1.
033110     WRITE EXCEPTION-REPORT-LINE FROM FIRST-HEADING-LINE
033120         AFTER ADVANCING TO-THE-TOP-OF-THE-PAGE.
033130     ADD 1 TO PAGE-NUMBER-COUNT.
033140     WRITE EXCEPTION-REPORT-LINE FROM SECOND-HEADING-LINE
033150         AFTER ADVANCING 2 LINES.
033160     WRITE EXCEPTION-REPORT-LINE FROM THIRD-HEADING-LINE
033170         AFTER ADVANCING 1 LINES.
033190     MOVE ZERO TO NUMBER-OF-LINES-PRINTED.
033190     MOVE SPACE-TWO-LINES TO PROPER-SPACING.
```

BEST COPY AVAILABLE

Creating a Sequential Master File


INPUT RECORDS
APPLICATION PROGRAM MASTER FILE
EXCEPTION REPORT

TM 2

Application Program Master File Information

FIELD	POSITION	NUMBER OF DIGITS	ATTRIBUTES
Program Identification	1 - 6	6	
Application Designation	1 - 2		Alphanumeric
Identification Number	3 - 6		Numeric
Program Description	7 - 26	20	Alphanumeric
Programmer Employee Number	27 - 32	6	Numeric
Estimated Review Hours	33 - 36	4	Numeric
Estimated Design Hours	37 - 40	4	Numeric
Estimated Code Hours	41 - 44	4	Numeric
Estimated Test Hours	45 - 48	4	Numeric
Estimated Implementation Hours	49 - 52	4	Numeric
Actual Review Hours	53 - 56	4	Numeric
Actual Design Hours	57 - 60	4	Numeric
Actual Code Hours	61 - 64	4	Numeric
Actual Test Hours	65 - 68	4	Numeric
Actual Implementation Hours	69 - 72	4	Numeric
Review Status Code	73	1	Alphanumeric
Design Status Code	74	1	Alphanumeric
Code Status Code	75	1	Alphanumeric
Test Status Code	76	1	Alphanumeric
Implementation Status Code	77	1	Alphanumeric

File Information:
Record Length – 77 characters
Blocking Factor – 20 records per block
Storage Medium – Direct Access Device
Organization – Sequential
Access – Sequential

Example of Check Totals
(Part 1 of 2)

Data Division

```
004030     05  TOTAL-ESTIMATED-HOURS-TRANSIN         PIC 9(4).
004040     05  ESTIMATED-TASK-HOURS-TRANSIN
004050         10  ESTIMATED-REVIEW-HOURS-TRANSIN    PIC 9(4).
004060         10  ESTIMATED-DESIGN-HOURS-TRANSIN    PIC 9(4).
004070         10  ESTIMATED-CODE-HOURS-TRANSIN      PIC 9(4).
004080         10  ESTIMATED-TEST-HOURS-TRANSIN      PIC 9(4).
004090         10  ESTIMATED-IMPLEM-HOURS-TRANSIN    PIC 9(4).
004100     05  ESTIMATED-HOURS-TABLE-TRANSIN REDEFINES
004110            ESTIMATED-TASK-HOURS-TRANSIN       PIC 9(4)
004120                                                OCCURS 5 TIMES.
```

```
007180     05  SUM-OF-ESTIMATED-HOURS-WORK PIC S9(5) VALUE ZERO
007190                                                USAGE IS COMP-3.

011060     05  SUM-ESTIMATED-HOURS-SUBSKRIPT  PIC S9(8) USAGE IS COMP
011070                                                  SYNC.
011080                                                  VALUE +6.
```

Example of Check Totals
(Part 2 of 2)

Procedure Divison

```
023050         IF ESTIMATED-HOURS-ARE-NUMERIC
023060            MOVE ZEROS TO SUM-OF-ESTIMATED-HOURS-WORK
023070            PERFORM D012-ADD-THE-HOURS
023080                  VARYING SUM-ESTIMATED-HOURS-SUBSCRIPT FROM 1 BY 1
023090                  UNTIL ALL-HOURS-HAVE-BEEN-ADDED
023100            IF SUM-OF-ESTIMATED-HOURS-WORK IS NOT =
023110                  TOTAL-ESTIMATED-HOURS-TRANSIN
023120               MOVE PROGRAM-IDENTIFICATION-TRANSIN TO
023130                     PROGRAM-ID-WORK-AREA
023140               MOVE TOTAL-EST-HOURS-FIELD-MSG TO
023150                     FIELD-IN-ERROR-WORK-AREA
023160               MOVE ESTIMATED-HOURS-SUM-ERROR-MSG TO
023170                     TYPE-OF-ERROR-WORK-AREA
023180               PERFORM U000-WRITE-EXCEPTION-REPORT
023190               MOVE 'NO ' TO IS-THE-RECORD-VALID.

024120 D012-ADD-THE-HOURS.
024130
024140     ADD ESTIMATED-HOURS-TABLE-TRANSIN
024150           (SUM-ESTIMATED-HOURS-SUBSCRIPT) TO
024160           SUM-OF-ESTIMATED-HOURS-WORK
```

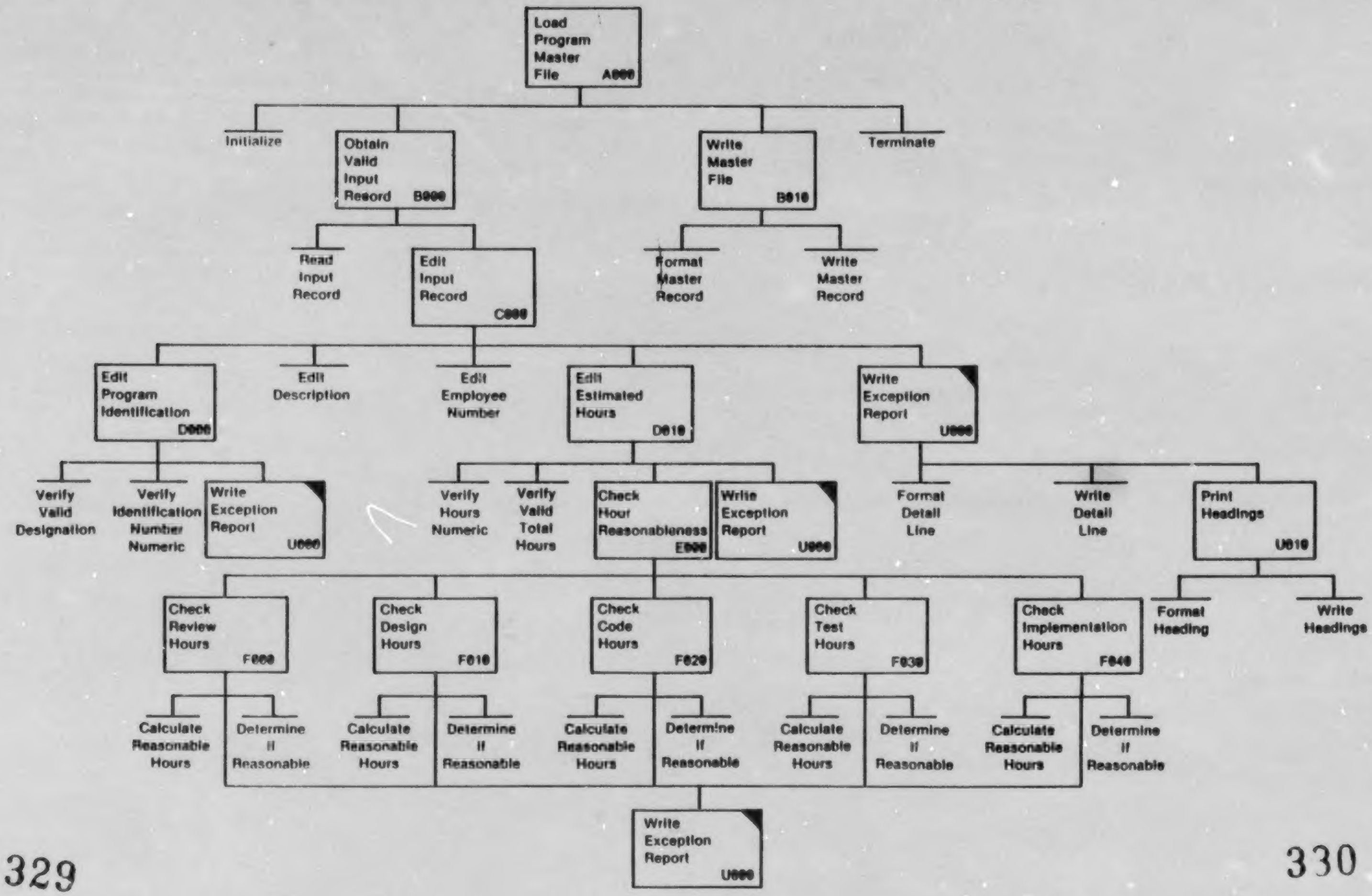
IPO Chart for Top-Level Module

Input/Processing/Output Chart

DATE: May 5	**PROGRAMMER:** Sandy Ash	**PROGRAM:** Load Program Master File
MODULE FUNCTION: Load the Program Master File	**REF:** A000	**MODULE NAME:** Load Program Master File

INPUT	PROCESSING	REF:	OUTPUT
1. Program	1. Initialize		1. Program
Transaction File	2. Obtain a valid input record	B000	Master File
	3. Write the master file	B010	
	4. Terminate		

Overall Hierarchy Chart


Load Program Master File A000
Initialize
Obtain Valid Input Record B000
Write Master File B010
Terminate
Read Input Record
Edit Input Record C000
Format Master Record
Write Master Record
Edit Program Identification D000
Edit Description
Edit Employee Number
Edit Estimated Hours D010
Write Exception Report U000
Verify Valid Designation
Verify Identification Number Numeric
Write Exception Report U000
Verify Hours Numeric
Verify Valid Total Hours
Check Hour Reasonableness E000
Write Exception Report U000
Format Detail Line
Write Detail Line
Print Headings U010
Check Review Hours F000
Check Design Hours F010
Check Code Hours F020
Check Test Hours F030
Check Implementation Hours F040
Format Heading
Write Headings
Calculate Reasonable Hours
Determine If Reasonable
Calculate Reasonable Hours
Determine If Reasonable
Calculate Reasonable Hours
Determine If Reasonable
Calculate Reasonable Hours
Determine If Reasonable
Calculate Reasonable Hours
Determine If Reasonable
Write Exception Report U000

Definition of Sequential File on Disk

Environment Division

```
003010     SELECT PROGRAM-MASTER-OUTPUT-FILE
003020           ASSIGN TO DA-S-PGMMSTR
```

Data Division

```
004050 FD  PROGRAM-MASTER-OUTPUT-FILE
004060       BLOCK CONTAINS 20 RECORDS
004070       RECORD CONTAINS 77 CHARACTERS
004080       LABEL RECORDS ARE STANDARD
004090       DATA RECORD IS PROGRAM-MASTER-OUTPUT-RECORD.
004100
004110 01  PROGRAM-MASTER-OUTPUT-RECORD.
004120     05  PROGRAM-IDENTIFICATION-MSTROUT.
004130         10  APPL-DESIGNATION-MSTROUT         PIC XX.
004140         10  IDENTIFICATION-NUMBER-MSTROUT    PIC X(4).
004150     05  PROGRAM-DESCRIPTION-MSTROUT          PIC X(20)
004160     05  PROGRAMMER-EMPL-NUMBER-MSTROUT       PIC X(6)
004170     05  ESTIMATED-TASK-HOURS-MSTROUT
004180         10  ESTIMATED-REVIEW-HOURS-MSTROUT   PIC 9(4).
004190         10  ESTIMATED-DESIGN-HOURS-MSTROUT   PIC 9(4).
004200         10  ESTIMATED-CODE-HOURS-MSTROUT     PIC 9(4).
005010         10  ESTIMATED-TEST-HOURS-MSTROUT     PIC 9(4).
005020         10  ESTIMATED-IMPLEM-HOURS-MSTROUT   PIC 9(4).
005030     05  ACTUAL-TASK-HOURS-MSTROUT.
005040         10  ACTUAL-REVIEW-HOURS-MSTROUT      PIC X(4).
005050         10  ACTUAL-DESIGN-HOURS-MSTROUT      PIC X(4).
005060         10  ACTUAL-CODE-HOURS-MSTROUT        PIC X(4).
005070         10  ACTUAL-TEST-HOURS-MSTROUT        PIC X(4).
005080         10  ACTUAL-IMPLEM-HOURS-MSTROUT      PIC X(4).
005090     05  TASK-STATUS-INDICATORS-MSTROUT
005100         10  REVIEW-STATUS-MSTROUT            PIC X.
005110         10  DESIGN-STATUS-MSTROUT            PIC X.
005120         10  CODE-STATUS-MSTROUT              PIC X.
005130         10  TEST-STATUS-MSTROUT              PIC X.
005140         10  IMPLEM-STATUS-MSTROUT            PIC X.
```

Example of the Use of Subscripted Fields (Part 1)

Data Division - Input Record

```
004040     05  ESTIMATED-TASK-HOURS-TRANSIN.
004050         10  ESTIMATED-REVIEW-HOURS-TRANSIN  PIC 9(4).
004060         10  ESTIMATED-DESIGN-HOURS-TRANSIN  PIC 9(4).
004070         10  ESTIMATED-CODE-HOURS-TRANSIN    PIC 9(4).
004080         10  ESTIMATED-TEST-HOURS-TRANSIN    PIC 9(4).
004090         10  ESTIMATED-IMPLEM-HOURS-TRANSIN  PIC 9(4).
004100     05  ESTIMATED-HOURS-TABLE-TRANSIN REDEFINES
004110           ESTIMATED-TASK-HOURS-TRANSIN      PIC 9(4).
004120                                              OCCURS 5 TIMES.
```

Example of the Use of Subscripted Fields (Part 2)

Data Division - Working - Storage

```
009020     05  ESTIMATED-HOURS-FIELD-MSGS.
009030         10  REVIEW-HOURS-FIELD-MSG   PIC X(30) VALUE
009040                                  'REVIEW HOURS                  '.
009050         10  DESIGN-HOURS-FIELD-MSG   PIC X(30) VALUE
009060                                  'DESIGN HOURS                  '.
009070         10  CODE-HOURS-FIELD-MSG     PIC X(30) VALUE
009080                                  'CODE HOURS                    '.
009090         10  TEST-HOURS-FIELD-MSG     PIC X(30) VALUE
009100                                  'TEST HOURS                    '.
009110         10  IMPLEM-HOURS-FIELD-MSG   PIC X(30) VALUE
009120                                  'IMPLEMENTATION HOURS          '.
009130     05  ALL-ESTIMATED-HOURS-FIELD-MSGS REDEFINES
009140           ESTIMATED-HOURS-FIELD-MSGS PIC X(30) OCCURS 5 TIMES.
```

```
011040     05  HOURS-NUMERIC-CHECK-SUBSCRIPT  PIC S9(8)  USAGE IS COMP
011050                                                   SYNC.
011060         88  ALL-HOURS-HAVE-BEEN-CHECKED           VALUE +6.
```

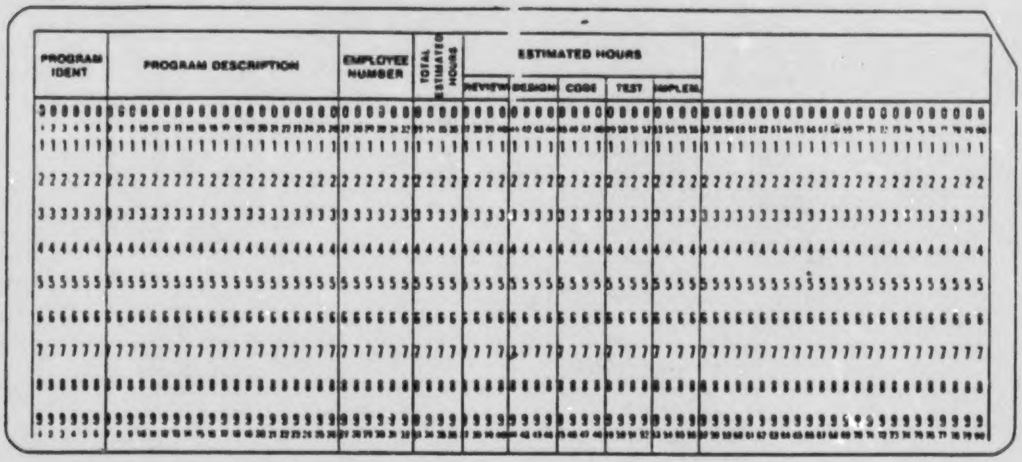
Example of the Use of Subscripted Fields (Part 3)

Procedure Division

```
022110     PERFORM D011-CHECK-FOR-NUMERIC-HOURS
022120           VARYING HOURS-NUMERIC-CHECK-SUBSCRIPT FROM 1 BY 1
022130           UNTIL ALL-HOURS-HAVE-BEEN-CHECKED.
```

```
024030 D011-CHECK-FOR-NUMERIC-HOURS.
024040
024050     IF ESTIMATED-HOURS-TABLE-TRANSIN
024060            (HOURS-NUMERIC-CHECK-SUBSCRIPT) IS NOT NUMERIC
024070         MOVE PROGRAM-IDENTIFICATION-TRANSIN TO
024080                PROGRAM-ID-WORK-AREA
024090         MOVE ALL-ESTIMATED-HOURS-FIELD-MSGS
024100                (HOURS-NUMERIC-CHECK-SUBSCRIPT) TO
024110                FIELD-IN-ERROR-WORK-AREA
024120         MOVE ESTIMATED-HOURS-ERROR-MSG TO
024130                TYPE-OF-ERROR-WORK-AREA
024140         PERFORM U000-WRITE-EXCEPTION-REPORT
024150         MOVE 'NO ' TO ARE-ESTIMATED-HOURS-NUMERIC
024160         MOVE 'NO ' TO IS-THE-RECORD-VALID.
```

Transaction File Input Record


PROGRAM IDENT
PROGRAM DESCRIPTION
EMPLOYEE NUMBER
TOTAL ESTIMATED HOURS
ESTIMATED HOURS
REVIEW
DESIGN
CODE
TEST
IMPLEM.

BEST COPY AVAILABLE

Example of Checking for Missing Data

```
019110     IF PROGRAM-DESCRIPTION-TRANSIN = SPACES
019120        MOVE PROGRAM-IDENTIFICATION-TRANSIN TO
019130              PROGRAM-ID-WORK-AREA
019140        MOVE DESCRIPTION-FIELD-MSG TO FIELD-IN-ERROR-WORK-AREA
019150        MOVE DESCRIPTION-ERROR-MSG TO TYPE-OF-ERROR-WORK-AREA
019160        PERFORM W000-WRITE-EXCEPTION-REPORT
019170        MOVE 'NO ' TO IS-THE-RECORD-VALID.
```

339

340

ASSIGNMENT SHEET 1: Edit Checks

Write a simple edit check sentence to check a field called HOURS-WORKED (PIC 9(3)V99) for numeric. Set a switch called INPUT-ERROR if the field is not numeric.

ASSIGNMENT SHEET 2: IPO Charts Summary

Using this format, make summarized IPO Charts for each module listed below.

	Module Name	Major Input	Major Processing	Major Outputs
1.	Edit Program Identification			
2.	Edit Estimated Hours			
3.	Write Exception Report			

SEQUENTIAL FILES I

UNIT IX

ANSWERS TO ASSIGNMENT SHEETS

Assignment Sheet 1

1. IF HOURS-WORKED IS NOT NUMERIC MOVE 'Y' TO INPUT-ERROR.

Assignment Sheet 2

Module Name	Major Input	Major Processing	Major Outputs
1. Edit Program Identification	Program Field	Check Program Field	Error Indicator
2. Edit Estimated Hours	Hours Field	Check for Errors	Error Indicator
3. Write Exception Report	Work Area	Write Report	Report Line

SEQUENTIAL FILES I

UNIT IX

EXAMINATION

1. Match the terms in Column B with correct phrase in Column A.

Column A	Column B
____ a. Prevents bad input data	1. Editing
____ b. Cross totals	2. Sequential file
____ c. Only digits in a field	3. Cross footing
____ d. In order from beginning to end	4. Reasonableness check
____ e. Predetermined value set by management	5. Check for numeric

2. True and false

T F a. The employee identification field in the sample program must not be numeric.

T F b. The driver module for the sample program is called the "Load Program Master File" module.

T F c. The purpose of the sample program is to create a sequential file of records about programs.

T F d. The programming specifications call for the fields containing actual hours to contain zeros.

3. Explain the three types of checks for numeric in COBOL.

4. Write the COBOL coding for a switch to indicate that the hours worked are greater than 100. Call the switch, ARE-HOURS-UNREASONABLE. Also write COBOL coding for a level 88 to accompany the switch to be called HOURS-ARE-UNREASONABLE.

SEQUENTIAL FILES I

UNIT IX

ANSWERS TO EXAMINATION

1. a. 1
   b. 3
   c. 5
   d. 2
   e. 4

2. a. F
   b. T
   c. T
   d. T

3. a. If the field is defined as alphanumeric, no sign is allowed. On some computers this check is not allowed.
   b. If the field is defined as numeric, no sign is allowed; only digits are allowed.
   c. If the field is defined as signed numeric, a sign is allowed with the digits.

4. 05 ARE-HOURS-UNREASONABLE PIC X(3) VALUE 'NO'.
   88 HOURS-ARE-UNREASONABLE VALUE 'YES'.

SORTING FOR SEQUENTIAL FILES

UNIT X

OBJECTIVES

After studying this unit, students should be able to apply sorting and merging techniques in sequential file oriented programs.

SPECIFIC OBJECTIVES

After studying this unit, the student should be able to:

1. Match terms and definitions.
2. Recognize correct statements on background materials on sorting and merging of sequential files.
3. List steps in sorting files and give the sample Environment, Data, and Procedure Entries for sort files.
4. Explain steps in the logic of the program's main procedure.
5. Recognize correct statements concerning the IPO and hierarchy charts for the sample merge program.
6. Recognize correct statements concerning the pseudocode for the sample program.
7. Recognize correct statements concerning the coding and listing of the sample program.
8. Apply the techniques learned in 1-7 to programming assignments.

SORTING FOR SEQUENTIAL FILES

UNIT X

SUGGESTED ACTIVITIES

I. Instructor

   A. Make transparencies.

   B. Provide students with objectives and discuss.

   C. Provide students with information sheets, assignment sheets, and text assignments and discuss.

   D. Give an examination.

II. Student

   A. Read objectives.

   B. Study information sheets and text as assigned by instructor.

   C. Complete assignment sheets and program assignments.

   D. Complete examination.

INSTRUCTIONAL MATERIALS

I. Unit content

   A. Objectives

   B. Information sheets

   C. Handouts

      1. HO 1: Program Specifications (Parts 1 and 2)

   D. Transparency masters

      1. TM 1: Example of Report Produced from Two Input Files

      2. TM 2: Format of Employee Identification File

      3. TM 3: Application Programmer Status Report Format

      4. TM 4: Environment Division Entries for Sort Files

   5. TM 5: Data Division File Definitions (Part I and 2)
   6. TM 6: Example of Sort Verb
   7. TM 7: Create Status Report Module
   8. TM 8: Hierarchy Chart
   9. TM 9: Example of Matching of Records Procedure
   10. TM 10: Pseudocode Specifications - Obtain Status Report Data
   11. TM 11: Source Listing (Part I and 2)

E. Examination

F. Answers to examination

II. References

A. Shelly, Gary B. and Thomas J. Cashman. "Sequential File Processing; Data Editing" in Advanced Structured COBOL: Program Design and File Processing. Brea, CA: Anaheim Publishing Co., 1981.

318

SORTING FOR SEQUENTIAL FILES

UNIT X

INFORMATION SHEETS

I. Terms and functional definitions

   A. **Sorting** - putting records into sequential order

   B. **Merging** - putting files which are in order into one combined file in the same sequence

   C. **SD** - sort definition clause

   D. **Sort's work file** - file used by the sort program to order records

II. Introduction to sorting and merging sequential files

   A. Introduction to sorting and merging

      1. Whenever two or more files are used as inputs to a program, they must be in a common sequence or order; thus, sorting is needed.

      2. The actual program controlled process of selectively combining all or part of records with common ID's is called merging.

      3. For example, a merge of an employee indentification file and a payroll file might result in the report as in TM 1, which combines data from both files.

      4. Difference between sorting and merging is that merges only work on files that are already in sequence.

   B. Sample application for sorting and merging

      1. Input files are the master file of application programs (from the previous unit) and an Employee Identification File (TM 2). These files will be used to produce an applications programmer status report (TM 3).

      2. Programming specifications for the sample program are given in HO 1.

         a. Items 2 and 3 of the specifications define the source of each data item.

         b. Note particularly items 5a and 5b in the specifications. Item 5a says that there will always be an entry for every programmer in the Employee Identification file, but there will be employees in the Employee Identification File who are not assigned to programs. A programmer may be assigned to more than one program so there will be multiple records for such a programmer in the Program Master File. These conditions are important as they will determine the logic of the program design.

III. Sorting files

A. Steps in sorting files

1. The program reads records from the program master file and puts them into a sort work file.

2. Invoke sort program to sort these records in the sort work file, which puts the records in an output file in sorted order.

B. Environment Division Entries (TM 4)

C. Data division entries

1. For the Sort Definition (SD) entry for definition of sort work files, see TM 5.

2. Notice that the work file is not blocked as it uses its own blocking factors. Otherwise, its definition is the same as an FD.

D. Procedure division entries - three file names

1. File-name-1 = the sort work file = SORT-WORK-FILE. (TM 6)

2. File-name-2 = the input file = PROGRAM-MASTER-INPUT-FILE.

3. File-name-3 = the output file = SORTED-PROGRAM-MASTER-FILE.

IV. Sample program design overview of the logic

A. Overview of the program's main procedure

1. Read a record from the program master file.

2. Read a record from the employee file.

3. Compare the employee number in both records.

4. If equal, then write a segment of the report.

5. If not equal, read records from employee file until equal; then process as in 4.

B. These steps assume that all programmers will have a record in the employee file. This is normally the case but in an abnormal situation, this could cause the program to fail. A more fail-proof technique will be introduced in the next unit.

V. Sample program design - IPO and Hierarchy Charts

A. The IPO chart for the top level module is given in TM 7.

B. The program's hierarchy chart is given in TM 8. This transparency gives the overall design, which is not complicated except in the matching of records.

VI. Sample program design

A. The sample code for the matching of records procedure is given in TM 9.

B. Although the code is short and simple, in reality it is performing the complicated function of matching employee records to program application records. Study it carefully.

C. Other code is reasonable and straightforward, but note that the module for the printing of the report uses group indication and subscripts. If unsure of these techniques, review them in previous unit.

VIII. Sample program implementation - the Coding and the Listing

A. Sample pseudocode of the Obtain Status Report Data module is given in TM 10.

B. The source listing of the sample program is given in TM 11. Study it for implementation techniques for similar problems.

HO 1

Program Specifications (Part 1 of 2)

Programming Specifications

SUBJECT Application Programmer Status Report	**DATE** May 2	**PAGE 1 OF 2**
TO Programmer	**FROM** Systems Analyst	

A program is to be written to create the Application Programmer Status Report. The format of the report and the input records are included as a part of this narrative. The program should be written to include the following processing.

1. The program should read the Program Master File and the Employee Identification File and create the Application Programmer Status Report.
2. The Employee Number, Program I.D., Program Description, Estimated Hours, and Actual Hours are taken directly from the Program Master File.
3. The Employee Name and Supervisor Name are taken directly from the Employee Identification File.
4. The records from the Program Master File and the Employee Identification File must be matched on the Employee Number. Thus, the Program Master File must be in an ascending sequence by Employee Number. The Employee Identification file is already stored in this sequence.
5. The following are the conditions under which the files are to be processed:
    a. Every record in the Program Master File has a matching record in the Employee Identification File; that is, for each Employee Number in the Program Master File, there is a corresponding record in the Employee Identification File.

    b. It is possible that a programmer is not assigned to a program. Therefore, there may be records in the Employee Identification File for which there are not matching records in the Program Master File.

    c. It is possible that a programmer is assigned to two or more programs. In this case, there will be more than one record in the Program Master File which matches a record in the Employee Identification File.

Program Specifications (Part 2 of 2)

Programming Specifications		
SUBJECT Application Programmer Status Report	**DATE** May 2	**PAGE 2 OF 2**
TO Programmer	**FROM** Systems Analyst	

d. All records which are contained in the Program Master File are to be reported on the Application Programmer Status Report. There is the possibility that one or more records in the Employee Identification File will not be contained on the report because the programmer is not assigned to a program.

6. The Application Programmer Status Report is to be printed according to the printer spacing chart which is included as a part of this narrative. The Employee Number, Employee Name, and Supervisor Name are to be group-indicated for each employee. The Program I.D. and Program Description are to be group-indicated for each program.

7. If the programmer is assigned to more than one program, the Program I.D., Program Description, the Program Tasks, Estimated Hours, Actual Hours, and Task Status fields should be printed for each program. The spacing should be as shown on the printer spacing chart.

8. Each programmer should be printed on a separate page. Therefore, whenever a different programmer is to be printed, new page headings should be printed.

9. The task status field should contain either the word "COMPLETE" or be blank. If the status field in the Program Master File record contains the value "C," then the task should be indicated as complete. Otherwise, the task status field should be blank on the report.

10. The Program Task field on the report will contain constants identifying each task.

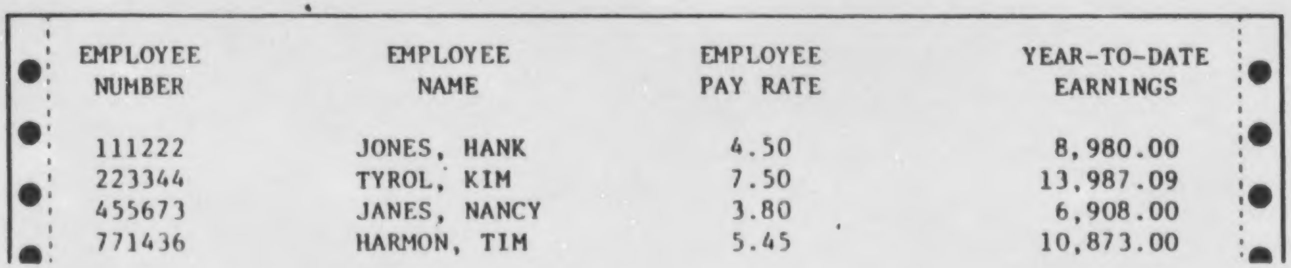
11. Totals should be taken for the Estimated Hours and the Actual Hours and should be printed on the report as illustrated on the printer spacing chart.

12. The program should be written in COBOL.

TM 1

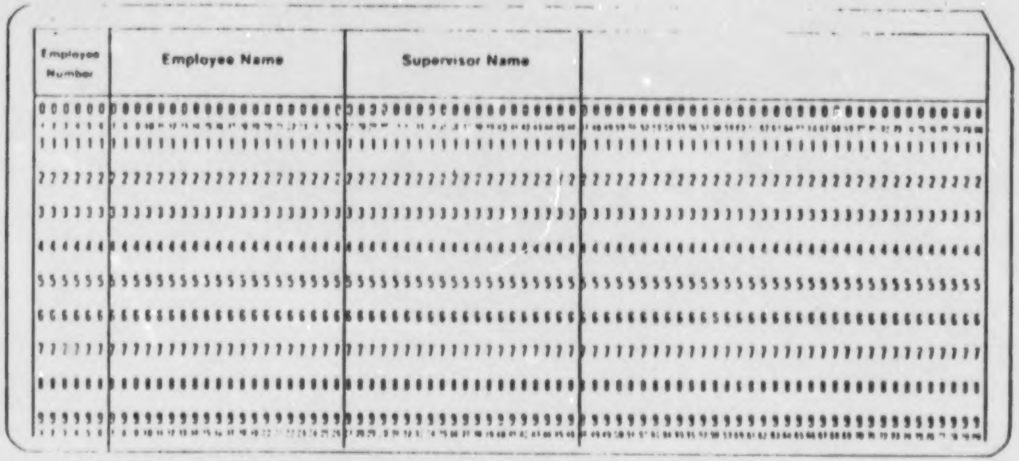
Example of Report Produced from Two Input Files

EXAMPLE REPORT



```
EMPLOYEE        EMPLOYEE          EMPLOYEE       YEAR-TO-DATE
 NUMBER           NAME            PAY RATE         EARNINGS

 111222       JONES, HANK           4.50           8,980.00
 223344       TYROL, KIM            7.50          13,987.09
 455673       JANES, NANCY          3.80           6,908.00
 771436       HARMON, TIM           5.45          10,873.00
```

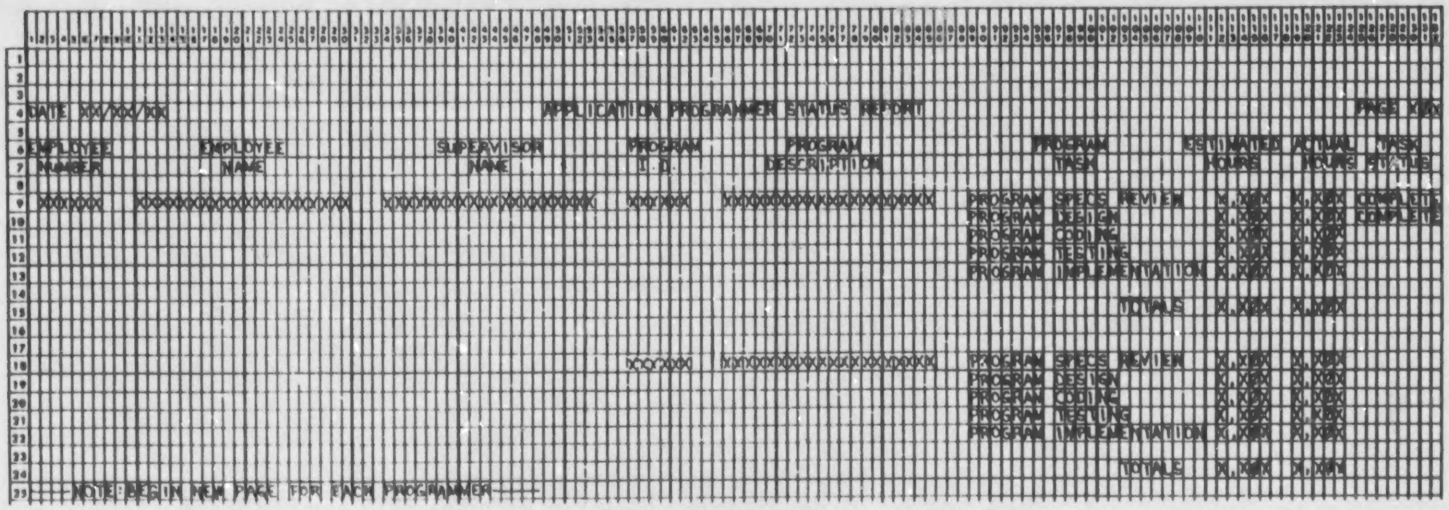
Format of Employee Identification File


Employee Number
Employee Name
Supervisor Name

TM 2

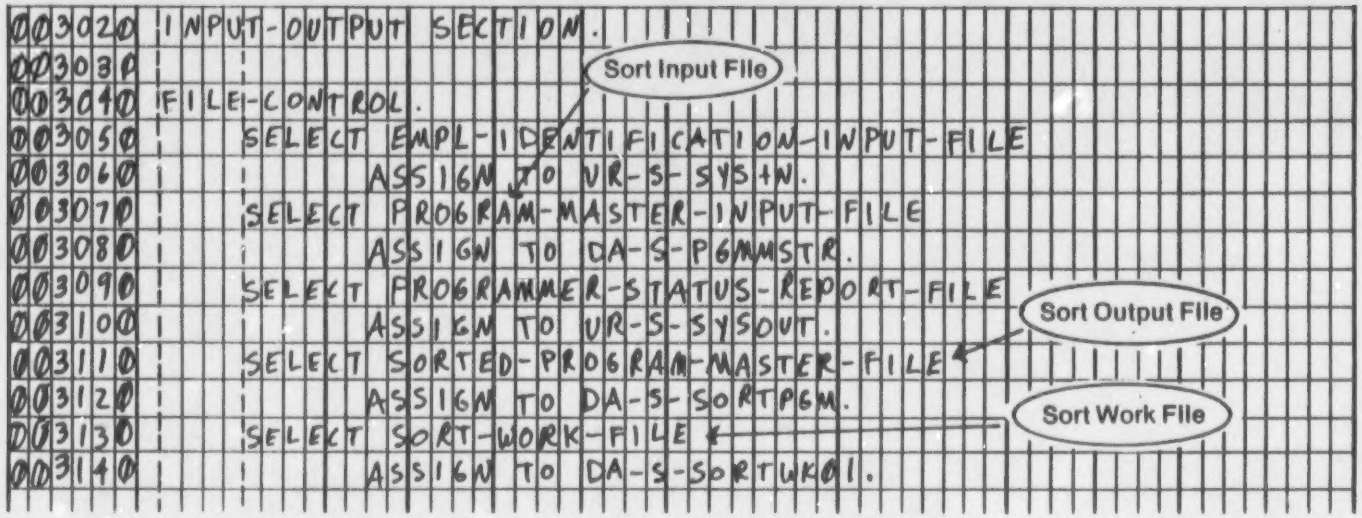
TM 3

Application Programmer Status Report Format



```
DATE XX/XX/XX                         APPLICATION PROGRAMMER STATUS REPORT                                        PAGE XXX

EMPLOYEE         EMPLOYEE               SUPERVISOR          PROGRAM         PROGRAM             PROGRAM          ESTIMATED  ACTUAL  TASK
 NUMBER            NAME                   NAME                I.D.        DESCRIPTION             TASK             HOURS    HOURS  STATUS

 XXXXXX   XXXXXXXXXXXXXXXXXXXX   XXXXXXXXXXXXXXXXXXXX   XXXXXX   XXXXXXXXXXXXXXXXXXXX  PROGRAM SPECS REVIEW     X,XXX    X,XXX  COMPLETE
                                                                                   PROGRAM DESIGN           X,XXX    X,XXX  COMPLETE
                                                                                   PROGRAM CODING           X,XXX    X,XXX
                                                                                   PROGRAM TESTING          X,XXX    X,XXX
                                                                                   PROGRAM IMPLEMENTATION   X,XXX    X,XXX

                                                                                                   TOTALS   X,XXX    X,XXX


                                                        XXXXXX   XXXXXXXXXXXXXXXXXXXX  PROGRAM SPECS REVIEW     X,XXX    X,XXX
                                                                                   PROGRAM DESIGN           X,XXX    X,XXX
                                                                                   PROGRAM CODING           X,XXX    X,XXX
                                                                                   PROGRAM TESTING          X,XXX    X,XXX
                                                                                   PROGRAM IMPLEMENTATION   X,XXX    X,XXX

                                                                                                   TOTALS   X,XXX    X,XXX
----NOTE: BEGIN NEW PAGE FOR EACH PROGRAMMER----
```

Environment Division Entries for Sort Files



```
003020 INPUT-OUTPUT SECTION.
003030
003040 FILE-CONTROL.
003050     SELECT EMPL-IDENTIFICATION-INPUT-FILE
003060           ASSIGN TO UR-S-SYSIN.
003070     SELECT PROGRAM-MASTER-INPUT-FILE
003080           ASSIGN TO DA-S-PGMMSTR.
003090     SELECT PROGRAMMER-STATUS-REPORT-FILE
003100           ASSIGN TO UR-S-SYSOUT.
003110     SELECT SORTED-PROGRAM-MASTER-FILE
003120           ASSIGN TO DA-S-SORTPGM.
003130     SELECT SORT-WORK-FILE
003140           ASSIGN TO DA-S-SORTWK01.
```

Sort Input File

Sort Output File

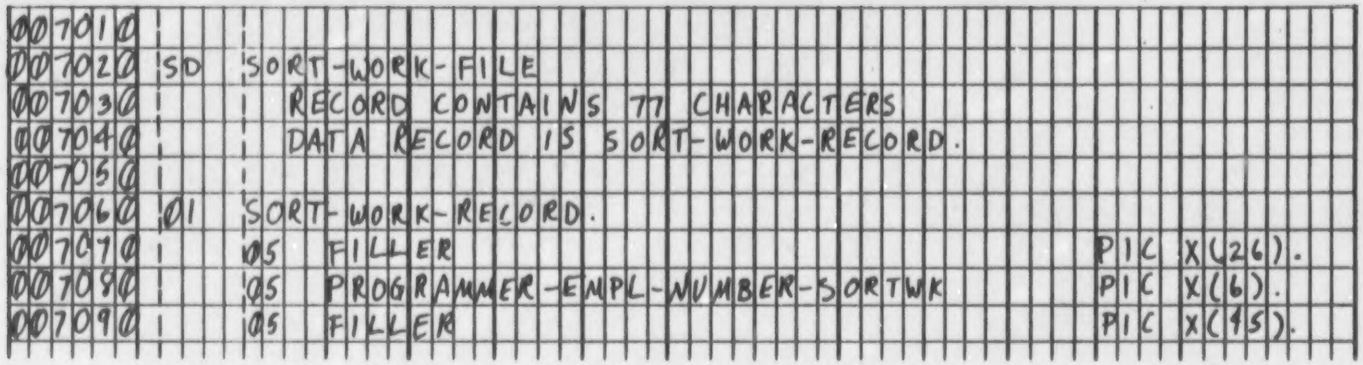
Sort Work File

TM 4

Data Division File Definitions
(Part 1 of 2)

```
005110 FD  PROGRAM-MASTER-INPUT-FILE
005120       BLOCK CONTAINS 20 RECORDS
005130       RECORD CONTAINS 77 CHARACTERS
005140       LABEL RECORDS ARE STANDARD
005150       DATA RECORD IS PROGRAM-MASTER-INPUT-RECORD.
005160
005170 01  PROGRAM-MASTER-INPUT-RECORD              PIC X(77).

006050 FD  SORTED-PROGRAM-MASTER-FILE
006060       BLOCK CONTAINS 20 RECORDS
006070       RECORD CONTAINS 77 CHARACTERS
006080       LABEL RECORDS ARE STANDARD
006090       DATA RECORD IS SORTED-MASTER-INPUT-RECORD.
006100
006110 01  SORTED-MASTER-INPUT-RECORD.
006120     05  PROGRAM-IDENTIFICATION-SORTMST       PIC X(6).
006130     05  PROGRAM-DESCRIPTION-SORTMST          PIC X(20).
006140     05  PROGRAMMER-EMPL-NUMBER-SORTMST       PIC X(6).
006150     05  ESTIMATED-TASK-HOURS-SORTMST OCCURS  5 TIMES
006160                                              PIC 9(4).
006170     05  ACTUAL-TASK-HOURS-SORTMST    OCCURS  5 TIMES
006180                                              PIC 9(4).
006190     05  TASK-STATUS-INDICATORS-SORTMST  OCCURS 5 TIMES
006200                                              PIC X.
006210         88  TASK-IS-COMPLETE                  VALUE  'C'
```

Data Division File Definitions
(Part 2 of 2)



```
007010
007020 SD  SORT-WORK-FILE
007030         RECORD CONTAINS 77 CHARACTERS
007040         DATA RECORD IS SORT-WORK-RECORD.
007050
007060 01  SORT-WORK-RECORD.
007070     05  FILLER                                  PIC X(26).
007080     05  PROGRAMMER-EMPL-NUMBER-SORTWK           PIC X(6).
007090     05  FILLER                                  PIC X(45).
```

```
[SD  file-name
     [; RECORD CONTAINS [integer-1 TO] integer-2 CHARACTERS]
    •[; DATA {RECORD IS  } data-name-1 [, data-name-2] ...]
             {RECORDS ARE}
```

Example of Sort Verb

```
022140     SORT SORT-WORK-FILE
022150          ASCENDING KEY PROGRAMMER-EMPL-NUMBER-SORTWK
022160          USING PROGRAM-MASTER-INPUT-FILE
022170          GIVING SORTED-PROGRAM-MASTER-FILE.
```

```
                    { ASCENDING  }
SORT file-name-1 ON { DESCENDING } KEY data-name-1 [, data-name-2] ...

USING file-name-2 [, file-name-3] ...

GIVING file-name-4
```

Create Status Report Module

Input/Processing/Output Chart

DATE: June 1

PROGRAMMER: Sandy Ash

PROGRAM: Application Programmer Status Report

MODULE FUNCTION: Create the Status Report

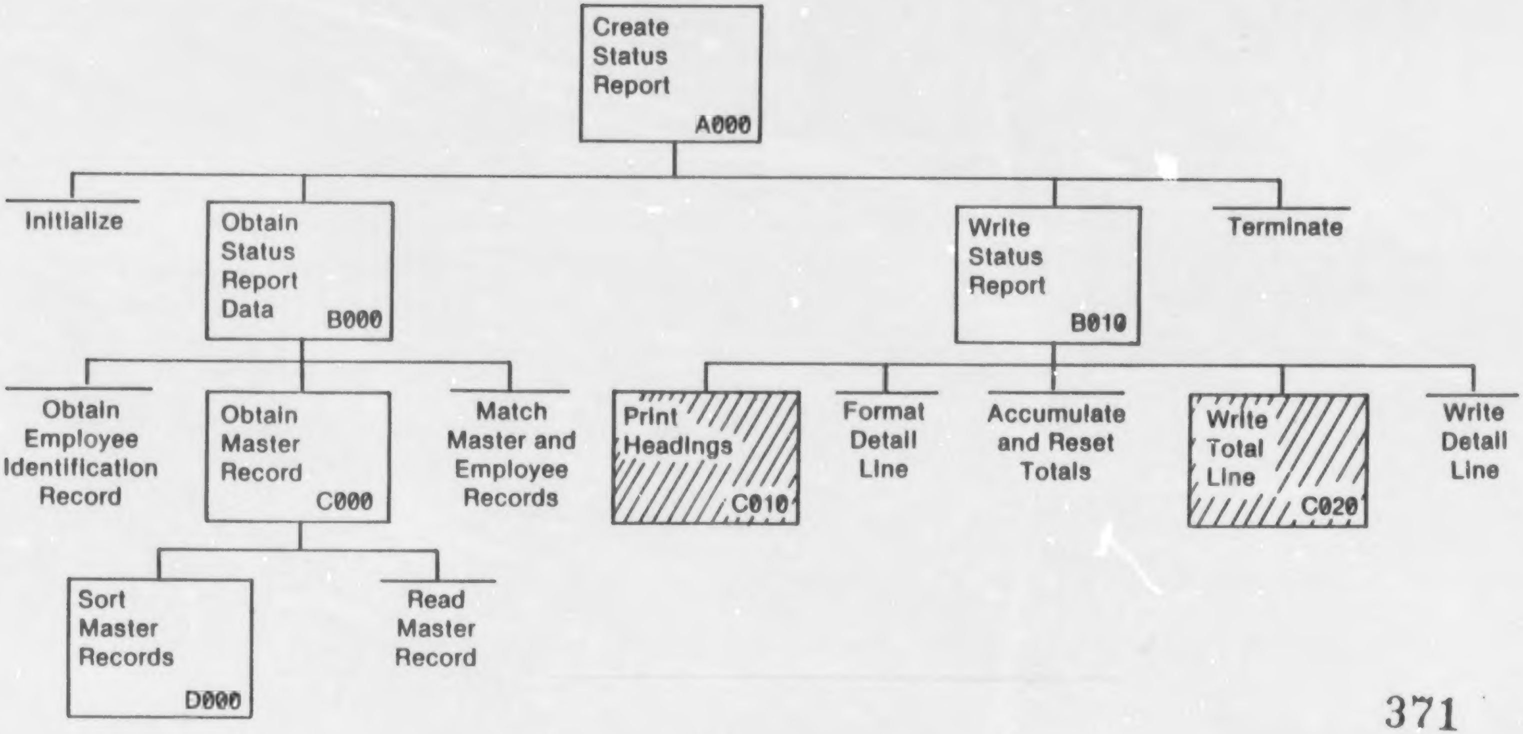
REF: A000

MODULE NAME: Create Status Report

INPUT	PROCESSING	REF:	OUTPUT
1. Program Master	1. Initialize		1. Application
File	2. Obtain status report data	B000	Programmer
2. Employee	3. Write status report	B010	Status Report
Identification File	4. Terminate		

TM 8

Hierarchy Chart

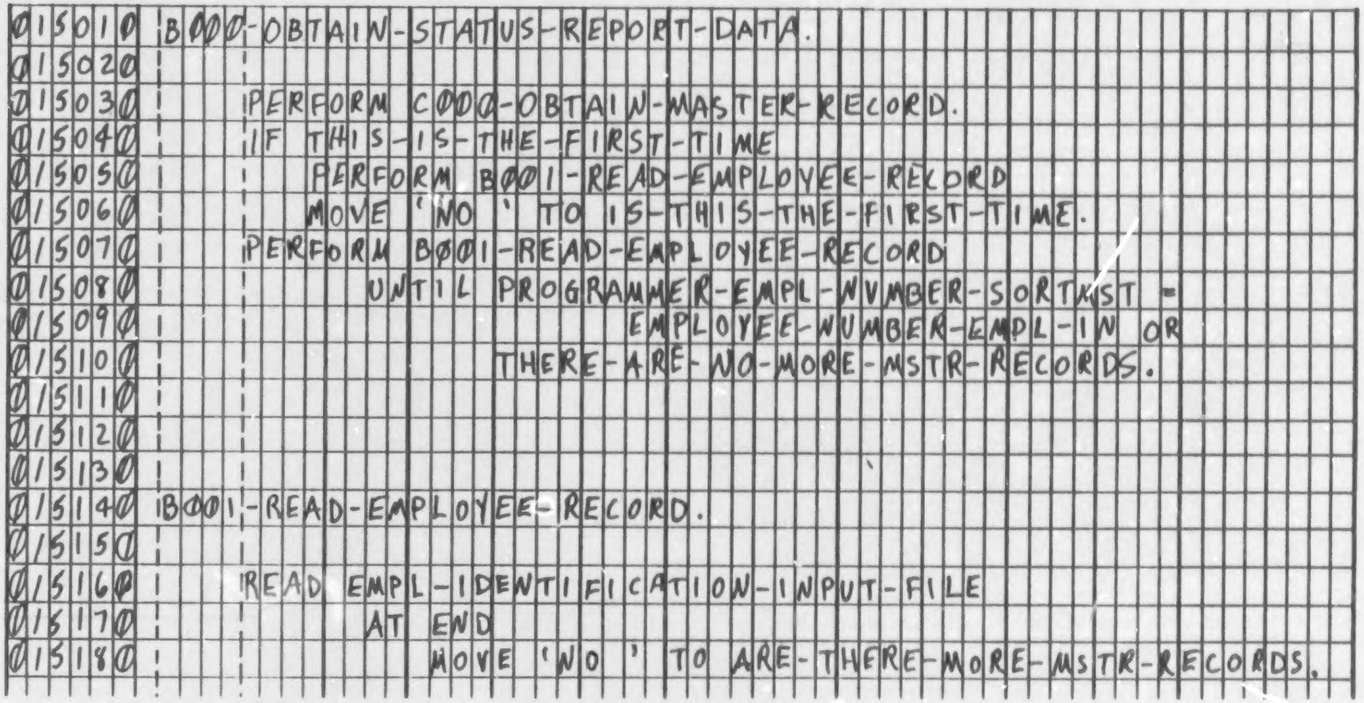


- Create Status Report A000
  - Initialize
  - Obtain Status Report Data B000
    - Obtain Employee Identification Record
    - Obtain Master Record C000
      - Sort Master Records D000
      - Read Master Record
    - Match Master and Employee Records
  - Write Status Report B010
    - Print Headings C010
    - Format Detail Line
    - Accumulate and Reset Totals
    - Write Total Line C020
    - Write Detail Line
  - Terminate

371

370

Example of Matching of Records Procedure



```
015010 B000-OBTAIN-STATUS-REPORT-DATA.
015020
015030     PERFORM C000-OBTAIN-MASTER-RECORD.
015040     IF THIS-IS-THE-FIRST-TIME
015050         PERFORM B001-READ-EMPLOYEE-RECORD
015060         MOVE 'NO ' TO IS-THIS-THE-FIRST-TIME.
015070     PERFORM B001-READ-EMPLOYEE-RECORD
015080            UNTIL PROGRAMMER-EMPL-NUMBER-SORTWST =
015090                         EMPLOYEE-NUMBER-EMPL-IN OR
015100                  THERE-ARE-NO-MORE-MSTR-RECORDS.
015110
015120
015130
015140 B001-READ-EMPLOYEE-RECORD.
015150
015160     READ EMPL-IDENTIFICATION-INPUT-FILE
015170            AT END
015180               MOVE 'NO ' TO ARE-THERE-MORE-MSTR-RECORDS.
```

TM 10

Pseudocode Specifications–Obtain Status Report Data

Pseudocode Specifications

DATE: June 1	**PROGRAMMER:** Jane Austin	**PROGRAM:** Application Programmer Status Report
MODULE FUNCTION: Obtain Data for the Status Report	**REF:** B000	**MODULE NAME:** Obtain Status Report Data

PSEUDOCODE

```
Obtain a program master record
IF this is the first time
   Read employee identification record
   Indicate not the first time
ENDIF
PERFORM UNTIL program master record =
         employee identification record or
         there are no more program master records
   Read employee identification file
ENDPERFORM
```

REF: C000

FILES, RECORDS, FIELDS REQUIRED

- First time indicator
- Employee number from program master record
- Employee number from employee identification file
- No more program master records indicator
- Employee identification file
- Employee file record area
  - Employee number
  - Employee name
  - Supervisor name

Source Listing (Part 1 of 2)

```
001010 IDENTIFICATION DIVISION.
001020
001030 PROGRAM-ID.     STATREPT.
001040 AUTHOR.         SHELLY AND CASHMAN.
001050 INSTALLATION.   ANAHEIM.
001060 DATE-WRITTEN.   06/01/78.
001070 DATE-COMPILED.  JUN 22,1978.
001080 SECURITY.       UNCLASSIFIED.
001090
001100*****************************************************************
001110*                                                               *
001120*  THIS PROGRAM CREATES THE APPLICATION PROGRAMMER STATUS       *
001130*  REPORT. THE OUTPUT FROM THE PROGRAM IS THE REPORT. THE INPUT*
001140*  TO THE PROGRAM CONSISTS OF THE APPLICATIONS PROGRAM MASTER   *
001150*  FILE (TO BE RESORTED INTO PROGRAMMER EMPLOYEE NUMBER         *
001160*  SEQUENCE) AND THE EMPLOYEE IDENTIFICATION FILE (ALREADY      *
001170*  SORTED IN PROGRAMMER EMPLOYEE NUMBER SEQUENCE). THE TWO      *
001180*  FILES ARE MATCHED ON PROGRAMMER EMPLOYEE NUMBER, AND WHEN A  *
001190*  MATCH OCCURS, DATA FROM BOTH RECORDS IS USED FOR THE REPORT.*
001200*  IT IS KNOWN PRIOR TO PROCESSING THE DATA IN THIS PROGRAM     *
002010*  THAT EVERY RECORD IN THE APPLICATIONS PROGRAM MASTER FILE    *
002020*  HAS A CORRESPONDING RECORD IN THE EMPLOYEE IDENTIFICATION    *
002030*  FILE. HOWEVER, THERE MAY NOT BE A RECORD IN THE PROGRAM      *
002040*  MASTER FILE FOR EVERY RECORD IN THE EMPLOYEE IDENTIFICATION *
002050*  FILE (I.E. SOME PROGRAMMERS MAY NOT BE ASSIGNED A PROGRAM)   *
002060*  OR THERE MAY BE TWO OR MORE RECORDS IN THE PROGRAM MASTER    *
002070*  FILE WHICH MATCH ONE RECORD IN THE EMPLOYEE IDENTIFICATION   *
002080*  FILE (I.E. A PROGRAMMER IS ASSIGNED TWO OR MORE PROGRAMS).   *
002090*                                                               *
002100*****************************************************************
002110
002120
002130
002140 ENVIRONMENT DIVISION.
002150
002160 CONFIGURATION SECTION.
002170
002180 SOURCE-COMPUTER. IBM-370.
002190 OBJECT-COMPUTER. IBM-370.
002200 SPECIAL-NAMES.   C01 IS TO-THE-TOP-OF-THE-PAGE.
003010
003020 INPUT-OUTPUT SECTION.
003030
003040 FILE-CONTROL.
003050     SELECT EMPL-IDENTIFICATION-INPUT-FILE
003060         ASSIGN TO UR-S-SYSIN.
003070     SELECT PROGRAM-MASTER-INPUT-FILE
003080         ASSIGN TO DA-S-PGMSTR.
003090     SELECT PROGRAMMER-STATUS-REPORT-FILE
003100         ASSIGN TO UR-S-SYSOUT.
003110     SELECT SORTED-PROGRAM-MASTER-FILE
003120         ASSIGN TO DA-S-SORTPGM.
003130     SELECT SORT-WORK-FILE
003140         ASSIGN TO DA-S-SORTWK01.
003160 DATA DIVISION.
003170
003180 FILE SECTION.
003190
003200 FD  EMPL-IDENTIFICATION-INPUT-FILE
004010       RECORD CONTAINS 80 CHARACTERS
004020       LABEL RECORDS ARE OMITTED
004030       DATA RECORD IS EMPL-IDENTIFICATION-INPUT-REC.
004040
004050 01  EMPL-IDENTIFICATION-INPUT-REC.
004060     05  EMPLOYEE-NUMBER-EMPL-IN                    PIC X(6).
004070     05  EMPLOYEE-NAME-EMPL-IN                      PIC X(20).
004080     05  SUPERVISOR-NAME-EMPL-IN                    PIC X(20).
004090     05  FILLER                                     PIC X(34).
004100
004110 FD  PROGRAM-MASTER-INPUT-FILE
004120       BLOCK CONTAINS 20 RECORDS
004130       RECORD CONTAINS 77 CHARACTERS
004140       LABEL RECORDS ARE STANDARD
004150       DATA RECORD IS PROGRAM-MASTER-INPUT-RECORD.
004160
004170 01  PROGRAM-MASTER-INPUT-RECORD                    PIC X(77).
004180
004190 FD  PROGRAMMER-STATUS-REPORT-FILE
004200       RECORD CONTAINS 133 CHARACTERS
005010       LABEL RECORDS ARE OMITTED
005020       DATA RECORD IS PROGRAMMER-STATUS-REPORT-LINE.
005030
005040 01  PROGRAMMER-STATUS-REPORT-LINE.
005050     05  CARRIAGE-CONTROL                           PIC X.
005060     05  FILLER                                     PIC X.
005070     05  EMPLOYEE-NUMBER-REPORT                     PIC X(6).
005080     05  FILLER                                     PIC X(3).
005090     05  EMPLOYEE-NAME-REPORT                       PIC X(20).
005100     05  FILLER                                     PIC X(3).
005110     05  SUPERVISOR-NAME-REPORT                     PIC X(20).
005120     05  FILLER                                     PIC X(3).
005130     05  PROGRAM-IDENTIFICATION-REPORT              PIC X(6).
005140     05  FILLER                                     PIC X(3).
005150     05  PROGRAM-DESCRIPTION-REPORT                 PIC X(20).
005160     05  FILLER                                     PIC X(3).
005170     05  PROGRAM-TASK-REPORT                        PIC X(22).
005180     05  FILLER                                     PIC X.
005190     05  ESTIMATED-HOURS-REPORT                     PIC Z,ZZ9.
005200     05  FILLER                                     PIC X(2).
006010     05  ACTUAL-HOURS-REPORT                        PIC Z,ZZ9.
006020     05  FILLER                                     PIC X.
006030     05  TASK-STATUS-REPORT                         PIC X(8).
006040
006050 FD  SORTED-PROGRAM-MASTER-FILE
006060       BLOCK CONTAINS 20 RECORDS
006070       RECORD CONTAINS 77 CHARACTERS
006080       LABEL RECORDS ARE STANDARD
006090       DATA RECORD IS SORTED-MASTER-INPUT-RECORD.
006100
006110 01  SORTED-MASTER-INPUT-RECORD.
006120     05  PROGRAM-IDENTIFICATION-SORTMST         PIC X(6).
006130     05  PROGRAM-DESCRIPTION-SORTMST            PIC X(20).
006140     05  PROGRAMMER-EMPL-NUMBER-SORTMST         PIC X(6).
006150     05  ESTIMATED-TASK-HOURS-SORTMST OCCURS 5 TIMES
006160                                                PIC 9(4).
006170     05  ACTUAL-TASK-HOURS-SORTMST    OCCURS 5 TIMES
006180                                                PIC 9(4).
006190     05  TASK-STATUS-INDICATORS-SORTMST OCCURS 5 TIMES
006200                                                PIC X.
006210         88  TASK-IS-COMPLETE                    VALUE 'C'.
007010
007020 SD  SORT-WORK-FILE
007030       RECORD CONTAINS 77 CHARACTERS
007040       DATA RECORD IS SORT-WORK-RECORD.
007050
007060 01  SORT-WORK-RECORD.
007070     05  FILLER                                 PIC X([illegible]).
007080     05  PROGRAMMER-EMPL-NUMBER-SORTWK          PIC X([illegible]
007090     05  FILLER                                 PIC X(45).
007100
007110 WORKING-STORAGE SECTION.
007120
007130 01  PROGRAM-INDICATORS.
007140     05  ARE-THERE-MORE-MSTR-RECORDS PIC XXX  VALUE 'YES'.
007150         88  THERE-ARE-NO-MORE-MSTR-RECORDS   VALUE 'NO '.
007160     05  IS-PGM-MASTER-FILE-SORTED   PIC XXX  VALUE 'NO '.
007170         88  PGM-MASTER-FILE-IS-NOT-SORTED    VALUE 'NO '.
007180     05  IS-THIS-THE-FIRST-TIME      PIC XXX  VALUE 'YES'.
007190         88  THIS-IS-THE-FIRST-TIME           VALUE 'YES'.
007200
007210 01  INPUT-RECORD-COMPARE-AREA.
007220     05  PREVIOUS-EMPLOYEE-NUMBER    PIC X(6) VALUE LOW-VALUES.
008010
008020 01  TOTAL-ACCUMULATORS                       USAGE IS COMP-3.
008030     05  ESTIMATED-HOURS-ACCUM   PIC S9(4)    VALUE ZERO.
008040     05  ACTUAL-HOURS-ACCUM      PIC S9(4)    VALUE ZERO.
008050
008060 01  PROGRAM-CONSTANTS.
008070     05  REPORT-PROGRAM-TASK-MSGS.
008080         10  FILLER              PIC X(22)    VALUE
008090                                      'PROGRAM SPECS REVIEW
008100         10  FILLER              PIC X(22)    VALUE
008110                                      'PROGRAM DESIGN
008120         10  FILLER              PIC X(22)    VALUE
008130                                      'PROGRAM CODING
008140         10  FILLER              PIC X(22)    VALUE
008150                                      'PROGRAM TESTING
008160         10  FILLER              PIC X(22)    VALUE
008170                                      'PROGRAM IMPLEMENTATION'
008180     05  ALL-REPORT-PROGRAM-TASK-MSGS REDEFINES
008190           REPORT-PROGRAM-TASK-MSGS           OCCURS 5 TIMES
008200                                 PIC X(22).
009010     05  TASK-STATUS-COMPLETE-MSG PIC X(8)    VALUE 'COMPLETE'.
009020
009030 01  PROGRAM-SUBSCRIPTS.
009040     05  PROGRAM-TASK-SUBSCRIPT  PIC S9(8)    USAGE IS COMP
009050                                              SYNC.
009060         88  ALL-TASKS-HAVE-BEEN-PRINTED      VALUE +6.
009070
009080 01  PRINTER-CONTROL.
009090     05  PROPER-SPACING          PIC 9.
009100     05  SPACE-ONE-LINE          PIC 9        VALUE 1.
009110     05  SPACE-TWO-LINES         PIC 9        VALUE 2.
009120     05  SPACE-THREE-LINES       PIC 9        VALUE 3.
009130     05  NUMBER-OF-LINES-PRINTED PIC S999     VALUE ZERO
009140                                              USAGE IS COMP-3.
009150     05  PAGE-SIZE               PIC 999      VALUE 40
009160                                              USAGE IS COMP-3.
009170     05  PAGE-NUMBER-COUNT       PIC S999     VALUE +1
009180                                              USAGE IS COMP-3.
009190         88  THIS-IS-THE-FIRST-PAGE           VALUE +1.
009200
010010 01  HEADING-LINES.
010020     05  FIRST-HEADING-LINE.
010030         10  CARRIAGE-CONTROL    PIC X.
010040         10  FILLER              PIC X(5)     VALUE 'DATE '.
010050         10  CURRENT-DATE-HDG1   PIC X(8).
010060         10  FILLER              PIC X(35)    VALUE SPACES.
010070         10  FILLER              PIC X(12)    VALUE
010080                                             'APPLICATION '.
010090         10  FILLER              PIC X(11)    VALUE 'PROGRAMMER '.
010100         10  FILLER              PIC X(7)     VALUE 'STATUS '.
010110         10  FILLER              PIC X(6)     VALUE 'REPORT'.
010120         10  FILLER              PIC X(40)    VALUE SPACES.
010130         10  FILLER              PIC X(5)     VALUE 'PAGE '.
010140         10  PAGE-NUMBER-HDG1    PIC ZZ9.
010150     05  SECOND-HEADING-LINE.
010160         10  CARRIAGE-CONTROL    PIC X.
010170         10  FILLER              PIC X(8)     VALUE 'EMPLOYEE'.
010180         10  FILLER              PIC X(8)     VALUE SPACES.
010190         10  FILLER              PIC X(8)     VALUE 'EMPLOYEE'.
010200         10  FILLER              PIC X(14)    VALUE SPACES.
011010         10  FILLER              PIC X(10)    VALUE 'SUPERVISOR'.
011020         10  FILLER              PIC X(8)     VALUE SPACES.
011030         10  FILLER              PIC X(7)     VALUE 'PROGRAM'.
011040         10  FILLER              PIC X(8)     VALUE SPACES.
011050         10  FILLER              PIC X(7)     VALUE 'PROGRAM'.
011060         10  FILLER              PIC X(16)    VALUE SPACES.
011070         10  FILLER              PIC X(7)     VALUE 'PROGRAM'.
011080         10  FILLER              PIC X(7)     VALUE SPACES.
011090         10  FILLER              PIC X(9)     VALUE 'ESTIMATED'.
011100         10  FILLER              PIC X        VALUE SPACE.
011110         10  FILLER              PIC X(6)     VALUE 'ACTUAL'.
011120         10  FILLER              PIC XX       VALUE SPACES.
011130         10  FILLER              PIC X(4)     VALUE 'TASK'.
011140         10  FILLER              PIC XX       VALUE SPACES.
```

BEST COPY AVAILABLE

Source Listing (Part 2 of 2)

```
    05  THIRD-HEADING-LINE.
        10  CARRIAGE-CONTROL      PIC X.
        10  FILLER                PIC X        VALUE SPACE.
        10  FILLER                PIC X(6)     VALUE 'NUMBER'.
        10  FILLER                PIC X(11)    VALUE SPACES.
        10  FILLER                PIC X(4)     VALUE 'NAME'.
        10  FILLER                PIC X(19)    VALUE SPACES.
        10  FILLER                PIC X(4)     VALUE 'NAME'.
        10  FILLER                PIC X(12)    VALUE SPACES.
        10  FILLER                PIC X(4)     VALUE 'I.D.'.
        10  FILLER                PIC X(8)     VALUE SPACES.
        10  FILLER                PIC X(11)    VALUE 'DESCRIPTION'.
        10  FILLER                PIC X(16)    VALUE SPACES.
        10  FILLER                PIC X(4)     VALUE 'TASK'.
        10  FILLER                PIC X(10)    VALUE SPACES.
        10  FILLER                PIC X(5)     VALUE 'HOURS'.
        10  FILLER                PIC X(4)     VALUE SPACES.
        10  FILLER                PIC X(5)     VALUE 'HOURS'.
        10  FILLER                PIC X        VALUE SPACE.
        10  FILLER                PIC X(6)     VALUE 'STATUS'.
        10  FILLER                PIC X        VALUE SPACE.

01  TOTAL-LINES.
    05  PROGRAM-TOTAL-LINE.
        10  CARRIAGE-CONTROL      PIC X.
        10  FILLER                PIC X(102)   VALUE SPACES.
        10  FILLER                PIC X(6)     VALUE 'TOTALS'.
        10  FILLER                PIC X(3)     VALUE SPACES.
        10  ESTIMATED-HOURS-TOTAL-LINE PIC Z,ZZ9.
        10  FILLER                PIC XX       VALUE SPACES.
        10  ACTUAL-HOURS-TOTAL-LINE PIC Z,ZZ9.
        10  FILLER                PIC X(9)     VALUE SPACES.
PROCEDURE DIVISION.

****************************************************************
*                                                              *
* THE FUNCTION OF THIS MODULE IS TO CREATE THE APPLICATION     *
* PROGRAMMER STATUS REPORT. IT IS ENTERED FROM AND EXITS TO    *
* THE OPERATING SYSTEM.                                        *
*                                                              *
****************************************************************

A000-CREATE-STATUS-REPORT.

    OPEN INPUT  EMPL-IDENTIFICATION-INPUT-FILE
         OUTPUT PROGRAMMER-STATUS-REPORT-FILE.
    PERFORM B000-OBTAIN-STATUS-REPORT-DATA.
    PERFORM A001-WRITE-AND-READ
         UNTIL THERE-ARE-NO-MORE-MSTR-RECORDS.
    CLOSE EMPL-IDENTIFICATION-INPUT-FILE
         PROGRAMMER-STATUS-REPORT-FILE.
    STOP RUN.



A001-WRITE-AND-READ.

    PERFORM B010-WRITE-STATUS-REPORT.
    PERFORM B000-OBTAIN-STATUS-REPORT-DATA.
****************************************************************
*                                                              *
*  THIS MODULE IS ENTERED TO OBTAIN DATA FOR THE APPLICATION   *
*  PROGRAMMER STATUS REPORT FROM TWO FILES - THE EMPLOYEE      *
*  IDENTIFICATION FILE AND THE PROGRAM MASTER FILE. IT IS      *
*  ENTERED FROM AND EXITS TO THE A000-CREATE-STATUS-REPORT     *
*  MODULE.                                                     *
*                                                              *
****************************************************************

B000-OBTAIN-STATUS-REPORT-DATA.

    PERFORM C000-OBTAIN-MASTER-RECORD.
    IF THIS-IS-THE-FIRST-TIME
       PERFORM B001-READ-EMPLOYEE-RECORD
       MOVE 'NO' TO IS-THIS-THE-FIRST-TIME.
    PERFORM B001-READ-EMPLOYEE-RECORD
         UNTIL PROGRAMMER-EMPL-NUMBER-SORTMST =
                    EMPLOYEE-NUMBER-EMPL-IN OR
               THERE-ARE-NO-MORE-MSTR-RECORDS.



B001-READ-EMPLOYEE-RECORD.

    READ EMPL-IDENTIFICATION-INPUT-FILE
         AT END
              MOVE 'NO ' TO ARE-THERE-MORE-MSTR-RECORDS.
****************************************************************
*                                                              *
*  THE FUNCTION OF THIS MODULE IS TO WRITE THE APPLICATION     *
*  PROGRAMMER STATUS REPORT. IT IS ENTERED FROM AND EXITS TO   *
*  THE A000-CREATE-STATUS-REPORT MODULE.                       *
*                                                              *
****************************************************************

B010-WRITE-STATUS-REPORT.

    IF PROGRAMMER-EMPL-NUMBER-SORTMST NOT =
            PREVIOUS-EMPLOYEE-NUMBER
       PERFORM C010-PRINT-HEADINGS
       MOVE SPACES TO PROGRAMMER-STATUS-REPORT-LINE
       MOVE PROGRAMMER-EMPL-NUMBER-SORTMST TO
               EMPLOYEE-NUMBER-REPORT
       MOVE EMPLOYEE-NAME-EMPL-IN TO EMPLOYEE-NAME-REPORT
       MOVE SUPERVISOR-NAME-EMPL-IN TO SUPERVISOR-NAME-REPORT
       MOVE PROGRAMMER-EMPL-NUMBER-SORTMST TO
               PREVIOUS-EMPLOYEE-NUMBER
    ELSE
       IF THIS-IS-THE-FIRST-PAGE OR
               NUMBER-OF-LINES-PRINTED = PAGE-SIZE OR
               IS GREATER THAN PAGE-SIZE
          PERFORM C010-PRINT-HEADINGS
          MOVE SPACES TO PROGRAMMER-STATUS-REPORT-LINE
          MOVE PROGRAMMER-EMPL-NUMBER-SORTMST TO
                  EMPLOYEE-NUMBER-REPORT
          MOVE EMPLOYEE-NAME-EMPL-IN TO EMPLOYEE-NAME-REPORT
          MOVE SUPERVISOR-NAME-EMPL-IN TO SUPERVISOR-NAME-REPORT.
    MOVE PROGRAM-IDENTIFICATION-SORTMST TO
            PROGRAM-IDENTIFICATION-REPORT.
    MOVE PROGRAM-DESCRIPTION-SORTMST TO
            PROGRAM-DESCRIPTION-REPORT.
    PERFORM B011-PRINT-EACH-TASK
            VARYING PROGRAM-TASK-SUBSCRIPT FROM 1 BY 1
            UNTIL ALL-TASKS-HAVE-BEEN-PRINTED.
    PERFORM C020-WRITE-TOTAL-LINE.
    MOVE ZEROS TO ESTIMATED-HOURS-ACCUM
                  ACTUAL-HOURS-ACCUM.
    MOVE SPACE-THREE-LINES TO PROPER-SPACING.
    MOVE SPACES TO PROGRAMMER-STATUS-REPORT-LINE.



B011-PRINT-EACH-TASK.

    MOVE ALL-REPORT-PROGRAM-TASK-MSGS (PROGRAM-TASK-SUBSCRIPT) TO
            PROGRAM-TASK-REPORT.
    MOVE ESTIMATED-TASK-HOURS-SORTMST (PROGRAM-TASK-SUBSCRIPT) TO
            ESTIMATED-HOURS-REPORT.
    MOVE ACTUAL-TASK-HOURS-SORTMST (PROGRAM-TASK-SUBSCRIPT) TO
            ACTUAL-HOURS-REPORT.
    IF TASK-IS-COMPLETE (PROGRAM-TASK-SUBSCRIPT)
       MOVE TASK-STATUS-COMPLETE-MSG TO TASK-STATUS-REPORT.
    WRITE PROGRAMMER-STATUS-REPORT-LINE
            AFTER PROPER-SPACING.
    ADD ESTIMATED-TASK-HOURS-SORTMST (PROGRAM-TASK-SUBSCRIPT) TO
            ESTIMATED-HOURS-ACCUM.
    ADD ACTUAL-TASK-HOURS-SORTMST (PROGRAM-TASK-SUBSCRIPT) TO
            ACTUAL-HOURS-ACCUM.
    ADD PROPER-SPACING TO NUMBER-OF-LINES-PRINTED.
    MOVE SPACE-ONE-LINE TO PROPER-SPACING.
    MOVE SPACES TO PROGRAMMER-STATUS-REPORT-LINE.
****************************************************************
*                                                              *
*  THE FUNCTION OF THIS MODULE IS TO OBTAIN A RECORD FROM THE  *
*  PROGRAM MASTER FILE. IT IS ENTERED FROM AND EXITS TO THE    *
*  B000-OBTAIN-STATUS-REPORT-DATA MODULE.                      *
*                                                              *
****************************************************************

C000-OBTAIN-MASTER-RECORD.

    IF PGM-MASTER-FILE-IS-NOT-SORTED
       PERFORM D000-SORT-MASTER-RECORDS
       MOVE 'YES' TO IS-PGM-MASTER-FILE-SORTED
       OPEN INPUT SORTED-PROGRAM-MASTER-FILE.
    READ SORTED-PROGRAM-MASTER-FILE
         AT END
              MOVE 'NO ' TO ARE-THERE-MORE-MSTR-RECORDS.
    IF THERE-ARE-NO-MORE-MSTR-RECORDS
       CLOSE SORTED-PROGRAM-MASTER-FILE.
****************************************************************
*                                                              *
*  THE FUNCTION OF THIS MODULE IS TO PRINT THE HEADINGS ON THE *
*  APPLICATION PROGRAMMER STATUS REPORT. IT IS ENTERED FROM AND*
*  EXITS TO THE B010-WRITE-STATUS-REPORT MODULE.               *
*                                                              *
****************************************************************

C010-PRINT-HEADINGS.

    MOVE CURRENT-DATE TO CURRENT-DATE-HDG1.
    MOVE PAGE-NUMBER-COUNT TO PAGE-NUMBER-HDG1.
    WRITE PROGRAMMER-STATUS-REPORT-LINE FROM FIRST-HEADING-LINE
            AFTER ADVANCING TO-THE-TOP-OF-THE-PAGE.
    ADD 1 TO PAGE-NUMBER-COUNT.
    WRITE PROGRAMMER-STATUS-REPORT-LINE FROM SECOND-HEADING-LINE
            AFTER ADVANCING 2 LINES.
    WRITE PROGRAMMER-STATUS-REPORT-LINE FROM THIRD-HEADING-LINE
            AFTER ADVANCING 1 LINES.
    MOVE SPACE-TWO-LINES TO PROPER-SPACING.
    MOVE ZERO TO NUMBER-OF-LINES-PRINTED.
****************************************************************
*                                                              *
*  THE FUNCTION OF THIS MODULE IS TO WRITE THE REPORT TOTAL    *
*  LINES. IT IS ENTERED FROM AND EXITS TO THE                  *
*  B010-WRITE-STATUS-REPORT MODULE.                            *
*                                                              *
****************************************************************

C020-WRITE-TOTAL-LINE.

    MOVE ESTIMATED-HOURS-ACCUM TO ESTIMATED-HOURS-TOTAL-LINE.
    MOVE ACTUAL-HOURS-ACCUM TO ACTUAL-HOURS-TOTAL-LINE.
    WRITE PROGRAMMER-STATUS-REPORT-LINE FROM PROGRAM-TOTAL-LINE
            AFTER ADVANCING 2 LINES.
****************************************************************
*                                                              *
*  THE FUNCTION OF THIS MODULE IS TO SORT THE PROGRAM MASTER   *
*  FILE IN AN ASCENDING SEQUENCE BY PROGRAMMER EMPLOYEE NUMBER.*
*  IT IS ENTERED FROM AND EXITS TO THE                         *
*  C000-OBTAIN-MASTER-RECORD MODULE.                           *
*                                                              *
****************************************************************

D000-SORT-MASTER-RECORDS.

    SORT SORT-WORK-FILE
         ASCENDING KEY PROGRAMMER-EMPL-NUMBER-SORTWK
         USING PROGRAM-MASTER-INPUT-FILE
         GIVING SORTED-PROGRAM-MASTER-FILE.
```

BEST COPY AVAILABLE

SORTING FOR SEQUENTIAL FILES

UNIT X

EXAMINATION

1. Match the following terms in Column B with their correct phrases in Column A.

	Column A		Column B
____ a.	Input files already in sequence	1.	Sorting
____ b.	Input files not in sequence	2.	Merging
____ c.	Temporary storage of records during a sort	3.	SD
____ d.	Defines a sort file	4.	Sort work's file

2. True and false

T F a. A merge works only on files which are already in sequence.

T F b. The programming specifications for the sample program state that there will always be a record in the Employee Identification File for every programmer.

T F c. In the IPO Chart for the CREATE STATUS REPORT module, the output is the updated Employee Identification File.

T F d. In the pseudocode for the OBTAIN STATUS REPORT DATA module, the employee identification file is read until a record is read whose ID matches that of program master record or until end of file.

T F e. In the sample coding for the OBTAIN STATUS REPORT DATA module, the data item named THIS-IS-THE-FIRST-TIME is not a level 88.

3. a. List two steps in sorting files.

   1.

   2.

   b. Write a typical SD for a file named SORT-WORK-FILE with record length of 120 characters. Do not bother with the 01 and 05 entries. The DATA RECORD is named SORT-WORK-RECORD.

4. In the program's main procedure, if the record ID in the program master file is not equal to the record ID in the employee file:

a. What happens?

b. Why?

378

SORTING FOR SEQUENTIAL FILES

UNIT X

ANSWERS TO EXAMINATION

1.
   a. 2
   b. 1
   c. 4
   d. 3

2.
   a. T
   b. T
   c. F
   d. T
   e. F

3.
   a.
      1. Read records from input file and place them in sort work file.
      2. Invoke sort program to sort work file and place sorted records in output file.

   b.
      1.
```
SD  SORT-WORK-FILE RECORD CONTAINS 120 CHARACTERS
        RECORD CONTAINS 120 CHARACTERS
        DATA RECORD IS SORT-WORK-RECORD.
```

4.
   a. The program continues to read records until an equal is found by passing the other records.
   b. These records are by-passed because they are for employees who did not program the particular application.

SEQUENTIAL FILE UPDATING

UNIT XI

UNIT OBJECTIVE

After studying this unit, the student should be able to apply sequential file updating techniques in COBOL programs.

SPECIFIC OBJECTIVES

After studying this unit, the student should be able to:

1. Match terms and definitions.
2. List major processing steps in a sequential file update program.
3. Explain detailed processing steps for each transaction type from the programming specifications.
4. Explain each of the three possible processing conditions in comparing master and transaction record keys.
5. Recognize correct statements concerning the sample program's design from IPO charts, hierarchy charts, pseudocode specifications, and program coding.
6. Apply preceding objectives in writing a sequential file update program.

380

SEQUENTIAL FILE UPDATING

UNIT XI

SUGGESTED ACTIVITIES

I. Instructor

  A. Make transparencies; prepare lectures, using this curriculum guide and supplemental materials.

  B. Provide students with objectives; discuss objectives.

  C. Make parallel assignments for reading and programming from the text(s).

  D. Provide students with information and assignment sheets; discuss these sheets.

  E. Provide students with additional supplementary information and exercises.

  F. Provide students with job sheet and make assignments.

  G. Give examination.

II. Student

  A. Read objectives.

  B. Study information sheets and text assignments.

  C. Complete assignment sheets and supplementary exercises.

  D. Complete assigned job sheet.

  E. Complete examination.

INSTRUCTIONAL MATERIALS

I. Unit Content

  A. Objectives

  B. Notes to the instructor

  C. Information sheets

D. Transparency masters

   1. TM 1: Flow Diagram of Sequential Update Application
   2. TM 2: Edit and Sort Programs Preceding a Typical Sequential File Update
   3. TM 3: Format of Program Master File
   4. TM 4: Format of Transaction Record
   5. TM 5: Program Specifications (Parts 1, 2, 3, and 4)
   6. TM 6: IPO Chart - Update Program Master File
   7. TM 7: Final Hierarchy Chart
   8. TM 8: Pseudocode for Update Program Master File
   9. TM 9: Pseudocode for Obtain New Master Data
   10. TM 10: Definition of Old and New Program Master Files
   11. TM 11: Source Listing (Parts 1, 2, and 3)

E. Assignment sheets

   1. AS 1: Introduction to Sequential File Updating
   2. AS 2: General Logic of Sequential File Update Programs
   3. AS 3: Flow chart of the General Logic of a Sequential Master File Update
   4. AS 4: Tracing the Flow of Records through Sequential File Update Logic

F. Answers to assignment sheets

G. Job Sheet 1: Writing a Complete Sequential File Update for Payroll (with an appendix)

   1. Figure JS 1-1: Payroll Application To Create, Update, and Print Payroll Master Files
   2. Figure JS 1-2: Prewritten Divisions for Payroll Applications
   3. Figure JS 1-3: Pseudocode Specifications for Program H1
   4. Figure JS 1-4: Input Files for Use in H1, H2, and H3 COBOL Programs

    5. Figure JS 1-5: Steps for Testing H1
    6. Figure JS 1-6: Output of H1
    7. Figure JS 1-7: Hierarchy Chart for Payroll Sequential File Update Program, H2
    8. Figure JS 1-8: Pseudocode for Payroll Sequential File Update Program, H2 (Parts 1 and 2)
    9. Figure JS 1-9: Output from Program H2
    10. Figure JS 1-10: Output from Program H3

H. Job Sheet 1: Appendix: Sample Application, Sequential Update Program for Bibliography

    1. Figure JS 1-11: Hierarchy Chart for Sample Bibliography Update Program
    2. Figure JS 1-12: Listing of Sample File Update Program
    3. Figure JS 1-13: File Layout and Listing of Old Master File For Sample Program
    4. Figure JS 1-14: Transaction Card Layout and List of Transactions Used in Sample Run for Sample Bibliography Update Program
    5. Figure JS 1-15: Output for Sample Run of Biblio - Update Program Combined with File Contents (Parts 1 and 2)
    6. Figure JS 1-16: Bibliography Update Transaction Report

I. Answers to Job Sheet 1

    1. Complete Program Solution to Job Sheet 1: H1, H2, and H3 Combined
    2. Data Files: Old Master and Updated Master
    3. Sample Correct Output for Program H2 of Job Sheet 1

J. Examination

K. Answers to examination

II. References

A. Shelly, Gary B. and Thomas J. Cashman. "Sequential File Updating" in Advanced Structured COBOL. Brea, CA: Anaheim Publishing Company, 1978.

B. Davis, et al. "Chapter 5, Sequential File Updating" in Elementary COBOL Programming. 2nd ed. New York: McGraw-Hill Publishing Co., in press.

SEQUENTIAL FILE UPDATING

UNIT XI

NOTES TO THE INSTRUCTOR

This unit contains a simple, but comprehensive payroll programming job for students to complete. This exercise is time-consuming for students and is of such a nature that many instructors use it as evidence of completion of a major portion of COBOL. Instructors also need to prepare the following files:

1. Data for the master file
2. Data for the transaction file
3. Prewritten COBOL divisions (to simplify coding efforts and aid grading)

Each of these files has been provided in printed form - one and two in the Answer to the Job Sheet and three as Figure JS 1-2 in the Job Sheet. These files need to be entered into your system for student use.

SEQUENTIAL FILE UPDATING

UNIT XI

INFORMATION SHEETS

I. Terms and functional definitions

- A. **Sequential file updating** - adding, changing, or deleting records in a sequential file
- B. **Addition transaction** - an entire record is added to a file
- C. **Change transaction** - fields within existing records are changed
- D. **Deletion transaction** - an entire record is removed from a file
- E. **Posting transaction** - similar to a change but generally applies to periodic updating of financial fields
- F. **File maintenance** - collective term for file updating
- G. **Program stub** - simulated module used to simplify early stages of program testing
- H. **Information hiding** - a programming principle which minimizes the amount of information handling designed into a program module

II. Introduction to sequential file updating

- A. Introduction
    - 1. Sequential files need to have their contents updated by adding, changing, or deleting records on a periodic basis.
    - 2. File maintenance is the collective term for updating of files.
    - 3. Additionally, files periodically have their financial values updated. This type of change is called posting and may not be considered part of file maintenance.
- B. Types of transactions in sequential file updating
    - 1. Four types of transactions are:
        - a. Addition transaction - add a new master record
        - b. Deletion transaction - delete a master record
        - c. Change transaction - change an existing master record

d. Posting transaction - change financial data on an existing master record

(NOTE: This type of transaction is not included in the sample program.)

2. Outputs from a sequential file update program are:

a. The updated (current) master file

b. A report of changes in the file

3. The four major processing steps in a sequential file update program are: (TM 1)

a. Read (obtain) records.

b. Compare records from the transaction file with the old master records.

c. Selectively update the contents of the old master file's records as controlled by type of transaction.

d. Write the output area onto the new (updated) master file.

4. The edit and sort programs (TM 2)

a. Normally precede a sequential file update program

b. Functions can also be combined into the sequential file update itself, as in this unit's sample program.

III. Overview of the sample program for sequential file updating

A. Layout of the master file - same application as in the two preceding units. Update of a Master File of Application Programs (TM 3) gives the layout of the master file.

B. Transaction file input - note column 80, the field which contains the type of transaction. (TM 4)

C. Programming specifications for the sample program (TM 5)

IV. Techniques in sequential file processing

A. The basic logical foundation is that both files are in the same sorted order. For example, once an ID of 783 is read, there can be no other ID in the rest of the file with a lower number. This is important for the logic of the techniques.

B. Detailed general processing steps are:

1. Read a record from both files—old master and transaction.

2. Compare the ID's in each record.

3. Based upon this comparison, three logical possibilities exist:

a. Transaction low - the Transaction ID is less than the Master ID, which means that this transaction can be an addition only.

b. Transaction equal - the Transaction ID is equal to the Master ID, which means that this transaction can be either a delete or change (also a posting, but not in this sample).

c. Transaction greater than - the Transaction ID is greater than the Master ID, which means there are no more transactions for this master record. In this case, the master is completely updated and can be written out as a new master record.

C. Transaction ID is less than the Master ID - "Transaction Low," steps in processing:

1. Check transaction code to make sure this is an "A" or addition-type transaction; if not, an "A" is an error

2. If it is an addition, then:

   a. Format the new record from the data in the transaction.

   b. Write the new master record. (Note - this is a simplifying assumption here. A more complex algorithm exists to allow changes to newly created records, which involves swapping the newly created master record with the master record in the work area. This allows changes to the newly created record in the same processing run.)

   c. Read the next transaction.

D. Transaction ID is equal to the Master ID - "Transaction Equal," steps in processing:

1. If a change trans (code = "C"), then:

   a. Any nonblank data fields are moved from the transaction to the old master.

   b. The next transaction is read.

2. If a delete trans (code = "D"), then:

   a. Read the next master record without writing the current one. This deletes it from the new master file.

   b. Read the next transaction.

3. A third possibility not covered in this program is that the transaction is a trailer record (as identified by HIGH-VALUES in its ID field). If it occurs, this is a signal to begin the end of processing routines. A different end of processing technique is used in this example.

E. Transaction ID is greater than the Master ID - "Transaction Greater." This means that processing is complete; there can be no more records for this particular master. The steps are:

388

1. Write the old master record onto the new master file as the new and updated master record.

2. Read another record from the old master file.

   (NOTE: do not read another transaction, as the one in the input area has not yet been processed.)

F. End of file processing

1. This is an unusually complex end of file processing, since either input file can be finished before the other, and the program is not finished until both files are completely processed.

2. Possible end of file conditions are:

   a. The old master file is finished but the transaction file is not. Any remaining records should be additions only. Any other records are errors. This is the same as the transaction ID less than the Master ID.

   b. The old master file is not finished but the transaction file is finished. Any records on the master should simply be copied to the new master without change. This is the same as the transaction ID greater than the Master ID.

   c. Both files are finished; this is the normal end of a job condition.

V. Sample program design - IPO and hierarchy charts

A. The IPO chart for the driver is shown in TM 6; detailed IPO charts are derived from it.

B. The final hierarchy chart is shown in TM 7.

VI. Sample program design - pseudocode specifications

A. Pseudocode for the driver (TM 8)

B. Pseudocode for the obtain new master module (TM 9)

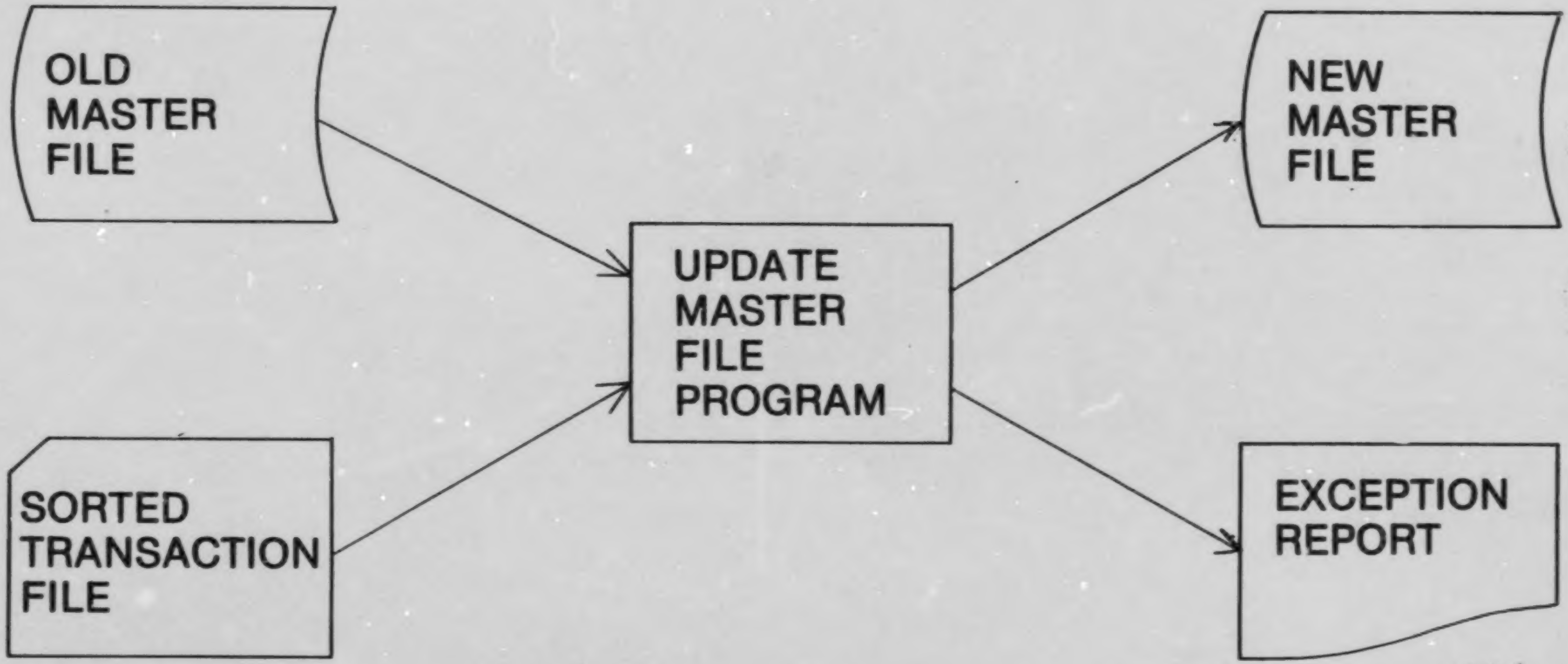
VII. Sample program coding and listing

A. Coding for defining the master files (TM 10)

B. Sample program listing (TM 11)

TM 1

Flow Diagram of Sequential Update Application



OLD MASTER FILE

SORTED TRANSACTION FILE

UPDATE MASTER FILE PROGRAM

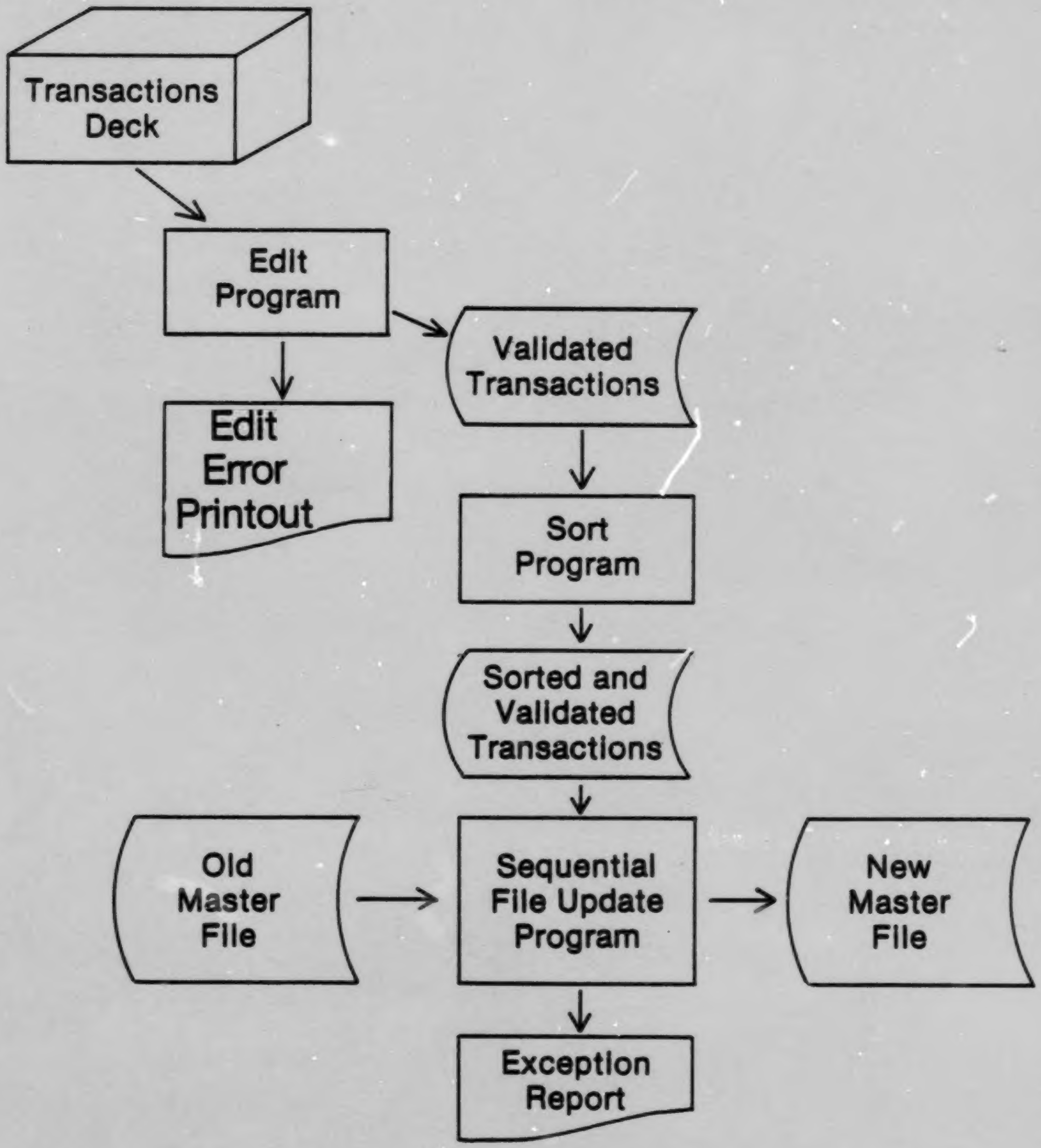
NEW MASTER FILE

EXCEPTION REPORT

361

TM 2

Edit and Sort Programs Preceding a Typical Sequential File Update


Transactions Deck
Edit Program
Validated Transactions
Edit Error Printout
Sort Program
Sorted and Validated Transactions
Old Master File
Sequential File Update Program
New Master File
Exception Report

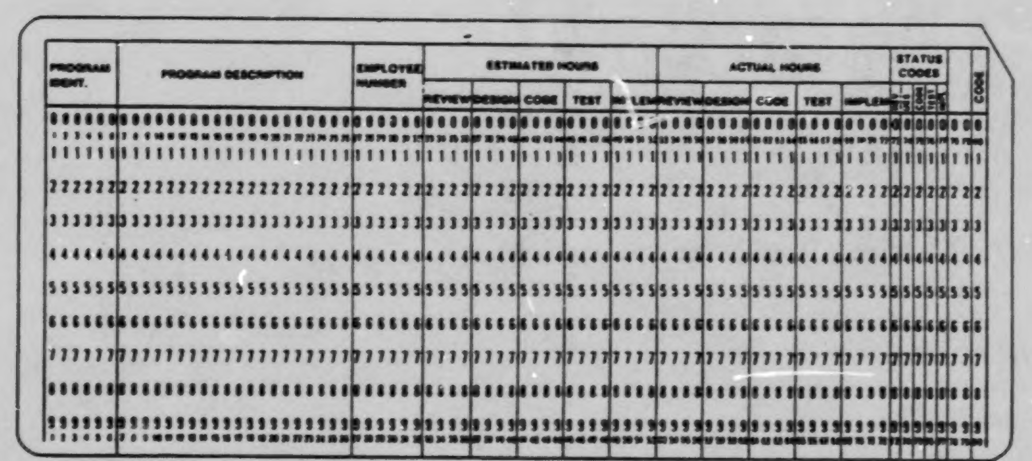
Format of Program Master File

FIELD	POSITION	NUMBER OF DIGITS	ATTRIBUTES
Program Identification	1-6	6	
Application Designation	1-2		Alphanumeric
Identification Number	3-6		Numeric
Program Description	7-26	20	Alphanumeric
Programmer Employee Number	27-32	6	Numeric
Estimated Review Hours	33-36	4	Numeric
Estimated Design Hours	37-40	4	Numeric
Estimated Code Hours	41-44	4	Numeric
Estimated Test Hours	45-48	4	Numeric
Estimated Implementation Hours	49-52	4	Numeric
Actual Review Hours	53-56	4	Numeric
Actual Design Hours	57-60	4	Numeric
Actual Code Hours	61-64	4	Numeric
Actual Test Hours	65-68	4	Numeric
Actual Implementation Hours	69-72	4	Numeric
Review Status Code	73	1	Alphanumeric
Design Status Code	74	1	Alphanumeric
Code Status Code	75	1	Alphanumeric
Test Status Code	76	1	Alphanumeric
Implementation Status Code	77	1	Alphanumeric

File Information:
Record Length - 77 characters
Blocking Factor - 20 records per block
Storage Medium - Direct Access Device
Organization - Sequential
Access - Sequential

TM 4

Format of Transaction Record


PROGRAM IDENT.
PROGRAM DESCRIPTION
EMPLOYEE NUMBER
ESTIMATED HOURS
REVIEW
DESIGN
CODE
TEST
IMPLEM
ACTUAL HOURS
REVIEW
DESIGN
CODE
TEST
IMPLEM
STATUS CODES
CODE

TM 5

Program Specifications (Part 1 of 4)

Programming Specifications

SUBJECT Update Application Program Master File	**DATE** June 10	Page 1 of 4
TO Programmer	**FROM** Systems Analyst	

Write a program to update the Application Program Master File. The format of the Application Program Master File, the Transaction File, which will be used to update the master file, and the Exception Report, to be created when there are errors found in the processing are included as a part of this narrative. The detailed processing is specified below.

1. The Program Master File is input to the program and contains the following fields: Program Identification, Program Description, Programmer Employee Number, Estimated Hours, Actual Hours, and Program Status Codes. The file is sorted in an ascending sequence by Program Identification.

2. The Transaction File contains transaction records, which indicate the type of updating which should occur to the Program Master File. The record can contain the following fields: Program Identification, Program Description, Programmer Employee Number, Estimated Hours, Actual Hours, Program Status Codes, and a Transaction Code.

3. Three types of processing can take place. A record can be added to the file, a record can be deleted from the master file, or a record in the Program Master File can be changed.

    a. Additions - A record is to be added to the Program Master File if the Transaction Code contains the value "A." When record is added to the Program Master File, The Program Identification, Program Description, Programmer Employee Number, and Estimated Hours are taken from the Transaction Record and are placed in the new Program Master Record. The Actual Hours in the new record are set to zero and the Program Status Codes fields are set to blank in the new record. An addition can only take place if there is no record on the Program Master File with the same Program Identification as a transaction record with a Transaction "A," this is an error and the addition cannot take place. Instead, an entry must be made on the Exception Report indicating an error.

Program Specifications (Part 2 of 4)

Programming Specifications

SUBJECT Update Applications Program Master File	**DATE** June 10	Page 2 of 4
TO Programmer	**FROM** Systems Analyst	

b. Deletions - A master record is to be deleted from the Program Master File if the Transaction Code in the Transaction Record Contains the value "D." In order for a deletion to take place, the Program Identification in the Transaction Record must be equal to the Program identification in the Program Master Record. When they are equal, the corresponding Master Record should not be written on the new Program Master File. The only fields required in the Transaction Record,whose function is to delete a master record are the Program Identification and the Transaction Code "D." If the Program Identification in the Transaction Record is not equal to a Program Identification in a Program Master Record and the Transaction Code is equal to "D," an error has occurred; and an entry should be made on the Exception Report.

c. Changes - Fields within a Program Master Record will be changed if the Transaction Code in the Transaction Record is equal to the value "C," and the Program Identification in the Transaction Record is equal to the Program Identification in the Program Master Record. There are three fields which can be changed in the Program Master Record - the Programmer Employee Number, the Actual Hours, and the Program Status Codes. In order for a change to a Master Record to take place, the Transaction Code in the Transaction Record must be equal to the value "C," and the Program Identification in the Transaction Record must equal the Program Identification in the Program Master Record.

When these conditions are met, then the following changes may take place:

1. If the Programmer Employee Number field in the Transaction Record contains data, then the value in the field will replace the value in the Program Master Record.
2. If any of the five Actual Hours fields (Review, Design, Code, Test, or Implement) contains data, that is, is not blank, then the value in the Transaction Record should be added to the appropriate field in the Program Master Record. For example, if the Actual Hours Code field was not blank in the Transaction Record, then the value in the Transaction Record should be added to the value in the Program Master Record.

Program Specifications (Part 3 of 4)

Programming Specifications

SUBJECT Update Application Program Master File	**DATE** June 10	PAGE 3 OF 4
TO Programmer	**FROM** Systems Analyst	

3. If any of the Status Codes in the Transaction Record (Review, Design, Code, Test, or Implement) contains the value "C," then the corresponding field in the Program Master Record should also be changed to the value "C." For example, if the Review Status Code field in the Transaction Record contained the value "C," then the value "C" should also be moved to the Review Status Code field in the Program Master Record.

If a Change record ("C" in Transaction Code) is read and there is no matching Program Master Record, then an entry should be made on the Exception Report indicating this error.

4. The only valid Transaction Codes are "A," "D," and "C." If any other code is found in a Transaction Record, the record is in error and should not be used in processing against the Program Master File. Instead, an entry should be made on the Exception Report.

5. The Exception Report is to contain the following information:
   a. Program Identification of the Transaction in error
   b. Transaction Code from the Transaction in error
   c. The Type of Error

The Types of Errors which can occur are:

a. Duplicate Master Record - This occurs when the Transaction Record indicates that an Addition is to take place, but there is already a master record in the Program Master File with a Program Identification equal to the one found in the Transaction Record.
b. Invalid Transaction Code - This occurs whenever the Transaction Code is not equal to "A," "D," or "C."
c. Non-Matching Master Record - This error occurs when a Transaction Code of "C" or "D" is found in the Transaction Record, but there is no Program Master Record with a Program Identification equal to the Program Identification in the Transaction Record.

Program Specifications (Part 4 of 4)

Programming Specifications

SUBJECT Update Application Program Master File	**DATE** June 10	PAGE 4 OF 4
TO Programmer	**FROM** Systems Analyst	

6. The Transaction File has been presorted in Program Identification sequence.

7. All fields in the Transaction Records,which may be used to add or change data within the Program Master File,have been previously edited. Therefore, there is no need to edit any input data in this program.

8. The program should be written in COBOL.

IPO Chart
Update Program Master File

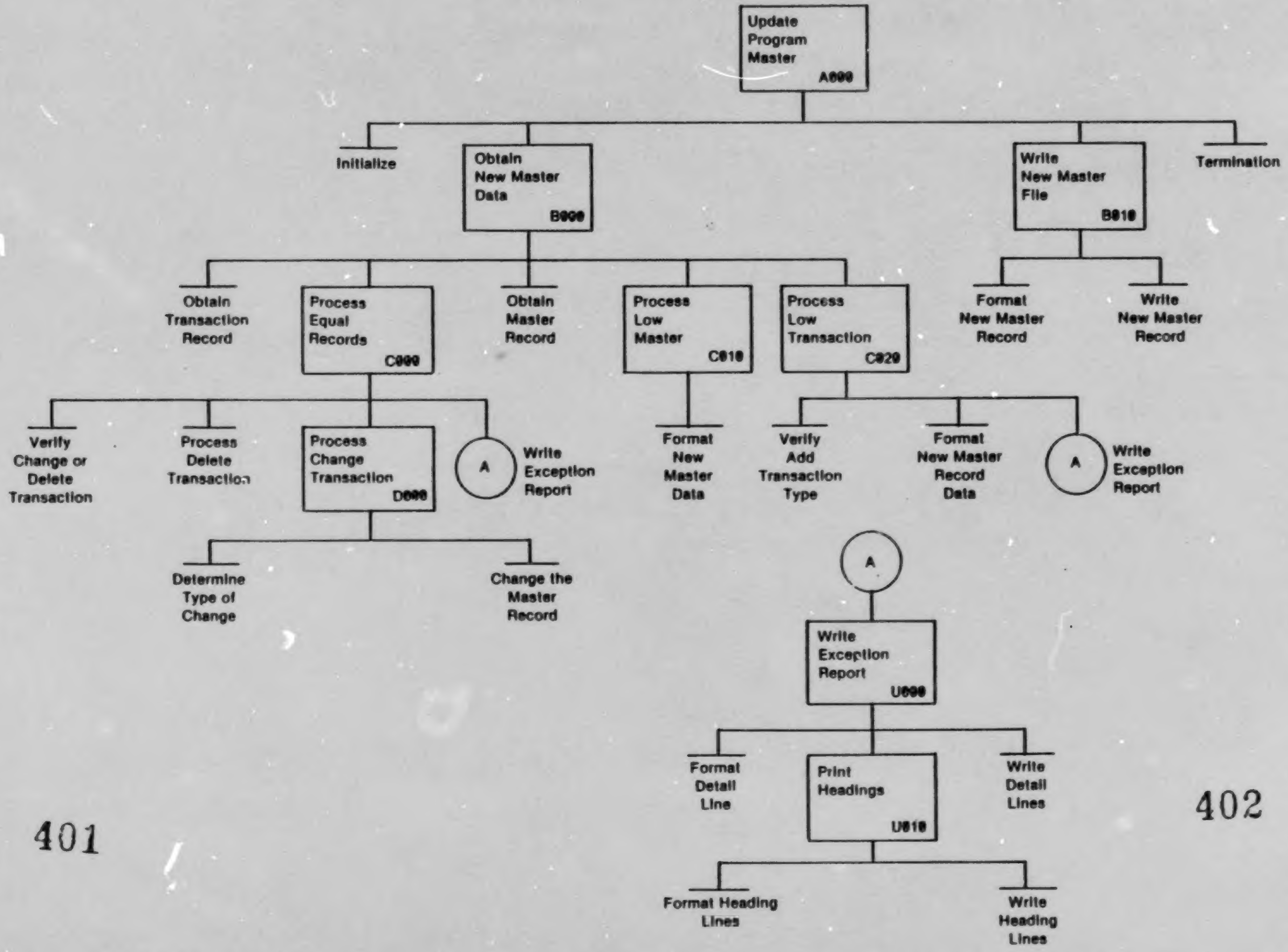
Input/Processing/Output Chart

DATE: June 16	**PROGRAMMER:** Jane Austin	**PROGRAM:** Program Master File Update
MODULE FUNCTION: Update the Program Master File	**REF:** A000	**MODULE NAME:** Update Program Master

INPUT	PROCESSING	REF:	OUTPUT
1. Old Program	1. Initialize		1. New Program
Master File	2. Obtain data for new master	B000	Master File
2. Update	3. Write new master file	B010	
Transaction File	4. Terminate		

369

Final Hierarchy Chart


Update Program Master A000
Initialize
Obtain New Master Data B000
Write New Master File B010
Termination
Obtain Transaction Record
Process Equal Records C000
Obtain Master Record
Process Low Master C010
Process Low Transaction C020
Format New Master Record
Write New Master Record
Verify Change or Delete Transaction
Process Delete Transaction
Process Change Transaction D000
A
Write Exception Report
Format New Master Data
Verify Add Transaction Type
Format New Master Record Data
A
Write Exception Report
Determine Type of Change
Change the Master Record
A
Write Exception Report U000
Format Detail Line
Print Headings U010
Write Detail Lines
Format Heading Lines
Write Heading Lines

370

401

402

Pseudocode for Update Program Master File

Pseudocode Specifications

DATE: June 17	PROGRAMMER: Jane Austin	PROGRAM: Program Master File Update
MODULE FUNCTION: Update the Program Master File	REF: A000	MODULE NAME: Update Program Master

```
PSEUDOCODE                                          REF:    FILES, RECORDS, FIELDS
                                                            REQUIRED

Open the files                                              Old program master file
Obtain new master data                              B000    New program master file
PERFORM UNTIL there are no more master records and          Update transaction file
      there are no more transaction records                 Update exception report file
   Write new master file                            B010    No more master records indicator
   Obtain new master data                           B000    No more transaction records indicator
ENDPERFORM                                                  Data obtained indicator
IF data obtained
   Write new master file                            B010
ENDIF
Close the files
Stop run
```

Pseudocode for Obtain New Master Data

Pseudocode Specifications

DATE: May 2	PROGRAMMER: Jane Austin	PROGRAM: Program Master File Update
MODULE FUNCTION: Obtain Data For the New Master	REF: B000	MODULE NAME: Obtain New Master Data

PSEUDOCODE / REF:

```
   Indicate no data obtained
   IF first time
      Read master
      Read transaction
      Indicate not the first time
   ENDIF
   PERFORM UNTIL data for new master is
         obtained or (there are no more master
         records and there are no more
         transaction records)
      Set add indicator to neutral
      Set delete indicator to neutral
      IF there is a master record and a
      transaction record                         C000
         IF master i.d. = transaction i.d.
            Process equal records
            IF valid delete
               Read master
            ENDIF
            Read transaction
      ELSE                                       C010
         IF master i.d.    transaction i.d.
            Process low master
            Indicate data obtained
            Read master record                   C020
         ELSE
            Process low transaction
            IF valid add
               Indicate data obtained
            ENDIF
            Read transaction
         ENDIF
      ENDIF
   ELSE
      IF there are no more master records        C020
         Process low transaction
         IF valid add
            Indicate data obtained
         ENDIF
         Read transaction
      ELSE
         Process low master                      C010
         Indicate data obtained
         Read master
      ENDIF
   ENDIF
ENDPERFORM
```

FILES, RECORDS, FIELDS REQUIRED

- Has data been obtained indicator
- First time indicator
- Input area for old master record
  - Program identification
  - Program description
  - Programmer employee number
  - Individual estimated hours (5)
    - Review hours
    - Design hours
    - Code hours
    - Test hours
    - Implementation hours
  - Actual hours (5)
    - Review hours
    - Design hours
    - Code hours
    - Test hours
    - Implementation hours
  - Status codes (5)
    - Review
    - Design
    - Code
    - Test
    - Implementation
- Input area for transaction record
  - Program identification
  - Program description
  - Programmer employee number
  - Individual estimated hours (5)
  - Actual hours (5)
  - Status codes (5)
  - Transaction code
- No more master records indicator
- No more transaction records indicator
- Valid delete record indicator
- Valid addition record indicator

404

Definition of Old and New Program Master Files

Old Master File

```
005070 FD  OLD-PROGRAM-MASTER-INPUT-FILE
005080         BLOCK CONTAINS 20 RECORDS
005090         RECORD CONTAINS 77 CHARACTERS
005100         LABEL RECORDS ARE STANDARD
005110         DATA RECORD IS OLD-PROGRAM-MASTER-INPUT-REC.
005120
005130 01  OLD-PROGRAM-MASTER-INPUT-REC.
005140     05  PROGRAM-IDENTIFICATION-OLDMST            PIC X(6).
005150     05  PROGRAM-DESCRIPTION-OLDMST               PIC X(20).
005160     05  PROGRAMMER-EMPL-NUMBER-OLDMST            PIC X(6).
005170     05  ESTIMATED-TASK-HOURS-OLDMST              PIC X(20).
005180     05  ACTUAL-TASK-HOURS-OLDMST.
005190         10  SINGLE-ACTUAL-HOUR-OLDMST            PIC S9(4)
005200                                                   OCCURS 5 TIMES.
005210     05  TASK-STATUS-INDICATORS-OLDMST.
005220         10  SINGLE-STATUS-INDICATOR-OLDMST       PIC X
005230                                                   OCCURS 5 TIMES.
```

New Master File

```
006030 FD  NEW-PROGRAM-MASTER-OUTPUT-FILE
006040         BLOCK CONTAINS 20 RECORDS
006050         RECORD CONTAINS 77 CHARACTERS
006060         LABEL RECORDS ARE STANDARD
006070         DATA RECORD IS NEW-PROGRAM-MASTER-OUTPUT-REC.
006080
006090 01  NEW-PROGRAM-MASTER-OUTPUT-REC.
006100     05  PROGRAM-IDENTIFICATION-NEWMST            PIC X(6).
006110     05  PROGRAM-DESCRIPTION-NEWMST               PIC X(20).
006120     05  PROGRAMMER-EMPL-NUMBER-NEWMST            PIC X(6).
006130     05  ESTIMATED-TASK-HOURS-NEWMST              PIC X(20).
006140     05  ACTUAL-TASK-HOURS-NEWMST                 PIC X(20).
006150     05  TASK-STATUS-INDICATORS-NEWMST            PIC X(5).
```

Source Listing (Part 1 of 3)

```
001010 IDENTIFICATION DIVISION.
001020
001030 PROGRAM-ID.     MSTRUPDT.
001040 AUTHOR.         SHELLY AND CASHMAN.
001050 INSTALLATION.   ANAHEIM.
001060 DATE-WRITTEN.   06/19/78.
001070 DATE-COMPILED.  AUG 17,1978.
001080 SECURITY.       UNCLASSIFIED.
001090
001100*****************************************************************
001110*                                                               *
001120*  THIS PROGRAM UPDATES THE APPLICATION PROGRAM MASTER FILE.    *
001130*  THE OUTPUT OF THE PROGRAM IS THE NEW, UPDATED APPLICATION    *
001140*  PROGRAM MASTER FILE AND AN EXCEPTION REPORT WHICH LISTS ANY  *
001150*  TRANSACTIONS WHICH CANNOT BE PROCESSED BY THE PROGRAM. THESE*
001160*  TRANSACTIONS INCLUDE 1) TRANSACTIONS WITH TRANSACTION CODES  *
001170*  OTHER THAN 'A' (ADD), 'D' (DELETE), OR 'C' (CHANGE);         *
001180*  2) ADD TRANSACTIONS WITH A CORRESPONDING MASTER ON THE OLD   *
001190*  MASTER FILE; 3) CHANGE OR DELETE TRANSACTIONS WITH NO        *
001200*  CORRESPONDING OLD MASTER RECORD. THE PROCESSING IN THE       *
002010*  PROGRAM CONSISTS OF THREE TYPES OF UPDATING WHICH CAN BE     *
002020*  PERFORMED ON THE APPLICATION PROGRAM MASTER FILE:            *
002030*      1) ADDITIONS - NEW RECORDS ARE ADDED TO THE MASTER FILE.*
002040*                     THE TRANSACTION CODE IS 'A'. THE DATA IN *
002050*                     THE NEW RECORD CONSISTS OF DATA IN THE   *
002060*                     TRANSACTION RECORD. THE ACTUAL HOURS IN  *
002070*                     THE NEW MASTER RECORD WILL BE SET TO ZERO*
002080*                     AND THE TASK STATUS CODES WILL BE SET    *
002090*                     EQUAL TO BLANKS.                         *
002100*      2) DELETIONS - RECORDS FROM THE MASTER FILE ARE DELETED.*
002110*                     THE TRANSACTION CODE IS 'D'. A DELETION  *
002120*                     TRANSACTION WILL CAUSE THE CORRESPONDING *
002130*                     MASTER RECORD TO NOT BE WRITTEN ON THE   *
002140*                     NEW MASTER FILE.                         *
002150*      3) CHANGES   - CHANGES ARE MADE TO EXISTING MASTER      *
002160*                     RECORDS AND THE CHANGED RECORDS ARE      *
002170*                     WRITTEN ON THE NEW MASTER FILE. THREE    *
002180*                     TYPES OF CHANGES CAN BE MADE - A CHANGE  *
002190*                     IN THE PROGRAMMER EMPLOYEE NUMBER,       *
002200*                     ADDITIONS TO THE ACTUAL HOURS USED FOR   *
003010*                     EACH PROGRAMMING TASK, AND CHANGING ANY  *
003020*                     OF THE TASK STATUS CODES FROM BLANK TO   *
003030*                     THE VALUE 'C' TO INDICATE THE TASK IS    *
003040*                     COMPLETE. THESE CHANGES WILL BE MADE IN  *
003050*                     THE MASTER RECORD WHENEVER THE           *
003060*                     CORRESPONDING FIELD IN THE TRANSACTION   *
003070*                     RECORD CONTAINS A NON-BLANK VALUE. ONE OR*
003080*                     MORE FIELDS CAN BE CHANGED BY ONE CHANGE *
003090*                     TRANSACTION RECORD AND ONE OR MORE CHANGE*
003100*                     TRANSACTION RECORDS CAN BE PROCESSED     *
003110*                     AGAINST A SINGLE OLD MASTER RECORD.      *
003120*                                                               *
003130*  ALL MASTER RECORDS AND TRANSACTION RECORDS ARE SORTED IN     *
003140*  AN ASCENDING SEQUENCE BY PROGRAM IDENTIFICATION, THE CONTROL*
003150*  FIELD. ALL DATA CONTAINED IN THE TRANSACTION RECORDS HAS     *
003160*  BEEN PREVIOUSLY EDITED AND WILL BE CORRECT WHEN USED AS      *
003170*  INPUT TO THIS PROGRAM.                                       *
003180*                                                               *
003190*****************************************************************
003200
004010
004020
004030 ENVIRONMENT DIVISION.
004040
004050 CONFIGURATION SECTION.
004060
004070 SOURCE-COMPUTER. IBM-370.
004080 OBJECT-COMPUTER. IBM-370.
004090 SPECIAL-NAMES.   C01 IS TO-THE-TOP-OF-THE-PAGE.
004100
004110 INPUT-OUTPUT SECTION.
004120
004130 FILE-CONTROL.
004140     SELECT OLD-PROGRAM-MASTER-INPUT-FILE
004150         ASSIGN TO DA-S-OLDMSTR.
004160     SELECT NEW-PROGRAM-MASTER-OUTPUT-FILE
004170         ASSIGN TO DA-S-NEWMSTR.
004180     SELECT UPDATE-TRANSACTION-INPUT-FILE
004190         ASSIGN TO UR-S-SYSIN.
004200     SELECT UPDATE-EXCEPTION-REPORT-FILE
005010         ASSIGN TO UR-S-SYSOUT.
005030 DATA DIVISION.
005040
005050 FILE SECTION.
005060
005070 FD  OLD-PROGRAM-MASTER-INPUT-FILE
005080       BLOCK CONTAINS 20 RECORDS
005090       RECORD CONTAINS 77 CHARACTERS
005100       LABEL RECORDS ARE STANDARD
005110       DATA RECORD IS OLD-PROGRAM-MASTER-INPUT-REC.
005120
005130 01  OLD-PROGRAM-MASTER-INPUT-REC.
005140     05  PROGRAM-IDENTIFICATION-OLDMST         PIC X(6).
005150     05  PROGRAM-DESCRIPTION-OLDMST            PIC X(20).
005160     05  PROGRAMMER-EMPL-NUMBER-OLDMST         PIC X(6).
005170     05  ESTIMATED-TASK-HOURS-OLDMST           PIC X(20).
005180     05  ACTUAL-TASK-HOURS-OLDMST.
005190         10  SINGLE-ACTUAL-HOUR-OLDMST         PIC S9(4)
005200                                                OCCURS 5 TIMES.
005210     05  TASK-STATUS-INDICATORS-OLDMST.
005220         10  SINGLE-STATUS-INDICATOR-OLDMST   PIC X
006010                                                OCCURS 5 TIMES.
006020
006030 FD  NEW-PROGRAM-MASTER-OUTPUT-FILE
006040       BLOCK CONTAINS 20 RECORDS
006050       RECORD CONTAINS 77 CHARACTERS
006060       LABEL RECORDS ARE STANDARD
006070       DATA RECORD IS NEW-PROGRAM-MASTER-OUTPUT-REC.
006080
006090 01  NEW-PROGRAM-MASTER-OUTPUT-REC.
006100     05  PROGRAM-IDENTIFICATION-NEWMST         PIC X(6).
006110     05  PROGRAM-DESCRIPTION-NEWMST            PIC X(20).
006120     05  PROGRAMMER-EMPL-NUMBER-NEWMST         PIC X(6).
006130     05  ESTIMATED-TASK-HOURS-NEWMST           PIC X(20).
006140     05  ACTUAL-TASK-HOURS-NEWMST              PIC X(20).
006150     05  TASK-STATUS-INDICATORS-NEWMST         PIC X(5).
006160
006170 FD  UPDATE-TRANSACTION-INPUT-FILE
006180       RECORD CONTAINS 80 CHARACTERS
006190       LABEL RECORDS ARE OMITTED
006200       DATA RECORD IS UPDATE-TRANSACTION-INPUT-REC.
007010
007020 01  UPDATE-TRANSACTION-INPUT-REC.
007030     05  PROGRAM-IDENTIFICATION-TRANS          PIC X(6).
007040     05  PROGRAM-DESCRIPTION-TRANS             PIC X(20).
007050     05  PROGRAMMER-EMPL-NUMBER-TRANS          PIC X(6).
007060     05  ESTIMATED-TASK-HOURS-TRANS            PIC X(20).
007090     05  ACTUAL-TASK-HOURS-TRANS.
007100         10  SINGLE-ACTUAL-HOUR-TRANS          PIC 9(4)
007110                                                OCCURS 5 TIMES.
007120     05  ACTUAL-TASK-HOURS-X-TRANS REDEFINES
007130           ACTUAL-TASK-HOURS-TRANS.
007140         10  SINGLE-ACTUAL-HOUR-X-TRANS        PIC X(4)
007150                                                OCCURS 5 TIMES.
007160     05  TASK-STATUS-INDICATORS-TRANS.
007170         10  SINGLE-STATUS-INDICATOR-TRANS     PIC X
007180                                                OCCURS 5 TIMES.
007190     05  FILLER                                PIC XX.
007200     05  TRANSACTION-CODE-TRANS                PIC X.
007210         88  THE-TRANSACTION-IS-AN-ADDITION     VALUE 'A'.
007220         88  THE-TRANSACTION-IS-A-DELETION      VALUE 'D'.
007230         88  THE-TRANSACTION-IS-A-CHANGE        VALUE 'C'.
007240
008010 FD  UPDATE-EXCEPTION-REPORT-FILE
008020       RECORD CONTAINS 133 CHARACTERS
008030       LABEL RECORDS ARE OMITTED
008040       DATA RECORD IS UPDATE-EXCEPTION-REPORT-LINE.
008050
008060 01  UPDATE-EXCEPTION-REPORT-LINE.
008070     05  CARRIAGE-CONTROL                      PIC X.
008080     05  FILLER                                PIC X(4).
008090     05  PROGRAM-IDENTIFICATION-REPORT         PIC X(6).
008100     05  FILLER                                PIC X(10).
008110     05  TRANSACTION-CODE-REPORT               PIC X.
008120     05  FILLER                                PIC X(11).
008130     05  TYPE-OF-ERROR-REPORT                  PIC X(30).
008140     05  FILLER                                PIC X(70).
008150
008160 WORKING-STORAGE SECTION.
008170
008180 01  PROGRAM-INDICATORS.
008190     05  ARE-THERE-MORE-MSTR-RECORDS      PIC XXX VALUE 'YES'.
008200         88  THERE-ARE-MORE-MSTR-RECORDS          VALUE 'YES'.
009010         88  THERE-ARE-NO-MORE-MSTR-RECORDS       VALUE 'NO '.
009020     05  ARE-THERE-MORE-TRAN-RECORDS      PIC XXX VALUE 'YES'.
009030         88  THERE-ARE-MORE-TRAN-RECORDS          VALUE 'YES'.
009040         88  THERE-ARE-NO-MORE-TRAN-RECORDS       VALUE 'NO '.
009050     05  HAS-DATA-BEEN-OBTAINED           PIC XXX VALUE 'NO '.
009060         88  DATA-HAS-BEEN-OBTAINED               VALUE 'YES'.
009070         88  DATA-HAS-NOT-BEEN-OBTAINED           VALUE 'NO '.
009080     05  IS-THIS-THE-FIRST-TIME           PIC XXX VALUE 'YES'.
009090         88  THIS-IS-THE-FIRST-TIME               VALUE 'YES'.
009100     05  WAS-IT-A-VALID-DELETE-RECORD     PIC XXX VALUE 'NO '.
009110         88 IT-WAS-A-VALID-DELETE-RECORD          VALUE 'YES'.
009120     05  WAS-IT-A-VALID-ADD-RECORD        PIC XXX VALUE 'NO '.
009130         88  IT-WAS-A-VALID-ADD-RECORD            VALUE 'YES'.
009140
009150 01  PROGRAM-WORK-AREAS.
009160     05  NEW-MASTER-WORK-AREA.
009170         10  PROGRAM-IDENTIFICATION-MSTWORK  PIC X(6).
009180         10  PROGRAM-DESCRIPTION-MSTWORK     PIC X(20).
009190         10  PROGRAMMER-EMPL-NUMBER-MSTWORK  PIC X(6).
009200         10  ESTIMATED-TASK-HOURS-MSTWORK    PIC X(20).
010010         10  ACTUAL-TASK-HOURS-MSTWORK       PIC X(20).
010020         10  TASK-STATUS-INDICATORS-MSTWORK  PIC X(5).
010030     05  EXCEPTION-REPORT-WORK-AREA.
010040         10  PROGRAM-IDENTIFICATION-RPTWORK  PIC X(6).
010050         10  TRANSACTION-CODE-RPTWORK        PIC X.
010060         10  TYPE-OF-ERROR-RPTWORK           PIC X(30).
010070
010080 01  PROGRAM-CONSTANTS.
010090     05  DUPLICATE-RECORD-ERRMSG         PIC X(30) VALUE
010100                                    'DUPLICATE MASTER RECORD       '.
010110     05  INVALID-TRANS-CODE-ERRMSG       PIC X(30) VALUE
010120                                    'INVALID TRANSACTION CODE      '.
010130     05  NON-MATCHING-MASTER-ERRMSG      PIC X(30) VALUE
010140                                    'NON-MATCHING MASTER RECORD    '.
010150     05  COMPLETED-CONSTANT              PIC X     VALUE 'C'.
010160
010170 01  PROGRAM-SUBSCRIPTS.
010180     05  ACTUAL-HOURS-ADD-SUBSCRIPT      PIC S9(8) USAGE IS COMP
010190                                                   SYNC.
```

BEST COPY

Source Listing (Part 2 of 3)

```
           88  ALL-ACTUAL-HOURS-ARE-ADDED                VALUE +6.
       05  STATUS-CODES-SUBSCRIPT          PIC S9(4) USAGE IS COMP
                                                     SYNC.
           88  ALL-STATUS-CODES-ARE-CHECKED              VALUE +6.

 01  PRINTER-CONTROL.
     05  PROPER-SPACING            PIC 9.
     05  SPACE-ONE-LINE            PIC 9          VALUE 1.
     05  SPACE-TWO-LINES           PIC 9          VALUE 2.
     05  NUMBER-OF-LINES-PRINTED PIC S999         VALUE ZERO
                                                  USAGE IS COMP-3.
     05  PAGE-SIZE                 PIC 999        VALUE 40
                                                  USAGE IS COMP-3.
     05  PAGE-NUMBER-COUNT         PIC S999       VALUE +1
                                                  USAGE IS COMP-3.
         88  THIS-IS-THE-FIRST-PAGE               VALUE +1.

 01  HEADING-LINES.
     05  FIRST-HEADING-LINE.
         10  CARRIAGE-CONTROL      PIC X.
         10  CURRENT-DATE-HDG1     PIC X(8).
         10  FILLER                PIC X(9)       VALUE SPACES.
         10  FILLER                PIC X(7)       VALUE 'UPDATE '.
         10  FILLER                PIC X(10)      VALUE 'EXCEPTION '.
         10  FILLER                PIC X(6)       VALUE 'REPORT'.
         10  FILLER                PIC X(10)      VALUE SPACES.
         10  FILLER                PIC X(5)       VALUE 'PAGE '.
         10  PAGE-NUMBER-HDG1      PIC ZZ9.
         10  FILLER                PIC X(74)      VALUE SPACES.
     05  SECOND-HEADING-LINE.
         10  CARRIAGE-CONTROL      PIC X.
         10  FILLER                PIC X(3)       VALUE SPACES.
         10  FILLER                PIC X(7)       VALUE 'PROGRAM'.
         10  FILLER                PIC X(6)       VALUE SPACES.
         10  FILLER                PIC X(11)      VALUE 'TRANSACTION'.
         10  FILLER                PIC X(13)      VALUE SPACES.
         10  FILLER                PIC X(4)       VALUE 'TYPE'.
         10  FILLER                PIC X(88)      VALUE SPACES.
     05  THIRD-HEADING-LINE.
         10  CARRIAGE-CONTROL      PIC X.
         10  FILLER                PIC X(14)      VALUE
                                                 'IDENTIFICATION'.
         10  FILLER                PIC X(5)       VALUE SPACES.
         10  FILLER                PIC X(4)       VALUE 'CODE'.
         10  FILLER                PIC X(15)      VALUE SPACES.
         10  FILLER                PIC X(8)       VALUE 'OF ERROR'.
         10  FILLER                PIC X(86)      VALUE SPACES.
 PROCEDURE DIVISION.

*****************************************************************
*                                                               *
*  THE FUNCTION OF THIS MODULE IS TO UPDATE THE PROGRAM MASTER  *
*  FILE. IT IS ENTERED FROM AND EXITS TO THE OPERATING SYSTEM.  *
*                                                               *
*****************************************************************

 A000-UPDATE-PROGRAM-MASTER.

     OPEN INPUT  OLD-PROGRAM-MASTER-INPUT-FILE
                 UPDATE-TRANSACTION-INPUT-FILE
          OUTPUT NEW-PROGRAM-MASTER-OUTPUT-FILE
                 UPDATE-EXCEPTION-REPORT-FILE.
     PERFORM B000-OBTAIN-NEW-MASTER-DATA.
     PERFORM A001-WRITE-AND-READ
         UNTIL THERE-ARE-NO-MORE-MSTR-RECORDS AND
               THERE-ARE-NO-MORE-TRAN-RECORDS.
     IF DATA-HAS-BEEN-OBTAINED
        PERFORM B010-WRITE-NEW-MASTER.
     CLOSE OLD-PROGRAM-MASTER-INPUT-FILE
           UPDATE-TRANSACTION-INPUT-FILE
           NEW-PROGRAM-MASTER-OUTPUT-FILE
           UPDATE-EXCEPTION-REPORT-FILE.
     STOP RUN.



 A001-WRITE-AND-READ.

     PERFORM B010-WRITE-NEW-MASTER.
     PERFORM B000-OBTAIN-NEW-MASTER-DATA.
*****************************************************************
*                                                               *
*  THE FUNCTION OF THIS MODULE IS TO OBTAIN THE DATA REQUIRED   *
*  FOR A NEW MASTER RECORD. IT IS ENTERED FROM AND EXITS TO THE*
*  A000-UPDATE-PROGRAM-MASTER MODULE.                           *
*                                                               *
*****************************************************************

 B000-OBTAIN-NEW-MASTER-DATA.

     MOVE 'NO ' TO HAS-DATA-BEEN-OBTAINED.
     IF THIS-IS-THE-FIRST-TIME
        PERFORM B002-READ-A-MASTER-RECORD
        PERFORM B003-READ-A-TRANSACTION-RECORD
        MOVE 'NO ' TO IS-THIS-THE-FIRST-TIME.
     PERFORM B001-COMPARE-INPUT-RECORDS
         UNTIL DATA-HAS-BEEN-OBTAINED OR
               (THERE-ARE-NO-MORE-MSTR-RECORDS AND
                THERE-ARE-NO-MORE-TRAN-RECORDS).

```

```

 B001-COMPARE-INPUT-RECORDS.

     MOVE SPACES TO WAS-IT-A-VALID-DELETE-RECORD.
     MOVE SPACES TO WAS-IT-A-VALID-ADD-RECORD.
     IF THERE-ARE-MORE-MSTR-RECORDS AND
           THERE-ARE-MORE-TRAN-RECORDS
        IF PROGRAM-IDENTIFICATION-OLDMST =
              PROGRAM-IDENTIFICATION-TRANS
           PERFORM C000-PROCESS-EQUAL-RECORDS
           IF IT-WAS-A-VALID-DELETE-RECORD
              PERFORM B002-READ-A-MASTER-RECORD
              PERFORM B003-READ-A-TRANSACTION-RECORD
           ELSE
              PERFORM B003-READ-A-TRANSACTION-RECORD
        ELSE
           IF PROGRAM-IDENTIFICATION-OLDMST IS LESS THAN
                 PROGRAM-IDENTIFICATION-TRANS
              PERFORM C010-PROCESS-LOW-MASTER
              MOVE 'YES' TO HAS-DATA-BEEN-OBTAINED
              PERFORM B002-READ-A-MASTER-RECORD
           ELSE
              PERFORM C020-PROCESS-LOW-TRANSACTION
              IF IT-WAS-A-VALID-ADD-RECORD
                 MOVE 'YES' TO HAS-DATA-BEEN-OBTAINED
                 PERFORM B003-READ-A-TRANSACTION-RECORD
              ELSE
                 PERFORM B003-READ-A-TRANSACTION-RECORD
     ELSE
        IF THERE-ARE-NO-MORE-MSTR-RECORDS
           PERFORM C020-PROCESS-LOW-TRANSACTION
           IF IT-WAS-A-VALID-ADD-RECORD
              MOVE 'YES' TO HAS-DATA-BEEN-OBTAINED
              PERFORM B003-READ-A-TRANSACTION-RECORD
           ELSE
              PERFORM B003-READ-A-TRANSACTION-RECORD
        ELSE
           PERFORM C010-PROCESS-LOW-MASTER
           MOVE 'YES' TO HAS-DATA-BEEN-OBTAINED
           PERFORM B002-READ-A-MASTER-RECORD.



 B002-READ-A-MASTER-RECORD.

     READ OLD-PROGRAM-MASTER-INPUT-FILE
          AT END
             MOVE 'NO ' TO ARE-THERE-MORE-MSTR-RECORDS.



 B003-READ-A-TRANSACTION-RECORD.

     READ UPDATE-TRANSACTION-INPUT-FILE
          AT END
             MOVE 'NO ' TO ARE-THERE-MORE-TRAN-RECORDS.
*****************************************************************
*                                                               *
*  THIS MODULE WRITES THE NEW MASTER RECORDS. IT IS ENTERED     *
*  FROM AND EXITS TO THE A000-UPDATE-PROGRAM-MASTER MODULE.     *
*                                                               *
*****************************************************************

 B010-WRITE-NEW-MASTER.

     MOVE PROGRAM-IDENTIFICATION-MSTWORK TO
             PROGRAM-IDENTIFICATION-NEWMST.
     MOVE PROGRAM-DESCRIPTION-MSTWORK TO
             PROGRAM-DESCRIPTION-NEWMST.
     MOVE PROGRAMMER-EMPL-NUMBER-MSTWORK TO
             PROGRAMMER-EMPL-NUMBER-NEWMST.
     MOVE ESTIMATED-TASK-HOURS-MSTWORK TO
             ESTIMATED-TASK-HOURS-NEWMST.
     MOVE ACTUAL-TASK-HOURS-MSTWORK TO
             ACTUAL-TASK-HOURS-NEWMST.
     MOVE TASK-STATUS-INDICATORS-MSTWORK TO
             TASK-STATUS-INDICATORS-NEWMST.
     WRITE NEW-PROGRAM-MASTER-OUTPUT-REC.
*****************************************************************
*                                                               *
*  THIS MODULE PROCESSES EQUAL MASTER AND TRANSACTION RECORDS.  *
*  IT IS ENTERED FROM AND EXITS TO THE B000-OBTAIN-NEW-MASTER-  *
*  DATA MODULE.                                                 *
*                                                               *
*****************************************************************

 C000-PROCESS-EQUAL-RECORDS.

     IF THE-TRANSACTION-IS-A-CHANGE
        PERFORM D000-PROCESS-CHANGE-TRANS
     ELSE
        IF THE-TRANSACTION-IS-A-DELETION
           MOVE 'YES' TO WAS-IT-A-VALID-DELETE-RECORD
        ELSE
           MOVE PROGRAM-IDENTIFICATION-TRANS TO
                   PROGRAM-IDENTIFICATION-RPTWORK
           MOVE TRANSACTION-CODE-TRANS TO
                   TRANSACTION-CODE-RPTWORK
           PERFORM C001-DETERMINE-TYPE-OF-ERROR
           PERFORM U000-WRITE-EXCEPTION-REPORT.

```

BEST COPY

BEST COPY AVAILABLE

Source Listing (Part 3 of 3)

```


 C001-DETERMINE-TYPE-OF-ERROR.

     IF THE-TRANSACTION-IS-AN-ADDITION
        MOVE DUPLICATE-RECORD-ERRMSG TO TYPE-OF-ERROR-RPTWORK
     ELSE
        MOVE INVALID-TRANS-CODE-ERRMSG TO TYPE-OF-ERROR-RPTWORK.
****************************************************************
*                                                              *
*  THE FUNCTION OF THIS MODULE IS TO PROCESS A LOW MASTER      *
*  RECORD. IT IS ENTERED FROM AND EXITS TO THE B000-OBTAIN-NEW-*
*  MASTER-DATA MODULE.                                         *
*                                                              *
****************************************************************

 C010-PROCESS-LOW-MASTER.

     MOVE PROGRAM-IDENTIFICATION-OLDMST TO
          PROGRAM-IDENTIFICATION-MSTWORK.
     MOVE PROGRAM-DESCRIPTION-OLDMST TO
          PROGRAM-DESCRIPTION-MSTWORK.
     MOVE PROGRAMMER-EMPL-NUMBER-OLDMST TO
          PROGRAMMER-EMPL-NUMBER-MSTWORK.
     MOVE ESTIMATED-TASK-HOURS-OLDMST TO
          ESTIMATED-TASK-HOURS-MSTWORK.
     MOVE ACTUAL-TASK-HOURS-OLDMST TO ACTUAL-TASK-HOURS-MSTWORK.
     MOVE TASK-STATUS-INDICATORS-OLDMST TO
          TASK-STATUS-INDICATORS-MSTWORK.
****************************************************************
*                                                              *
*  THE FUNCTION OF THIS MODULE IS TO PROCESS A LOW TRANSACTION *
*  RECORD. IT IS ENTERED FROM AND EXITS TO THE                 *
*  B000-OBTAIN-NEW-MASTER-DATA MODULE.                         *
*                                                              *
****************************************************************

 C020-PROCESS-LOW-TRANSACTION.

     IF THE-TRANSACTION-IS-AN-ADDITION
        MOVE PROGRAM-IDENTIFICATION-TRANS TO
             PROGRAM-IDENTIFICATION-MSTWORK
        MOVE PROGRAM-DESCRIPTION-TRANS TO
             PROGRAM-DESCRIPTION-MSTWORK
        MOVE PROGRAMMER-EMPL-NUMBER-TRANS TO
             PROGRAMMER-EMPL-NUMBER-MSTWORK
        MOVE ESTIMATED-TASK-HOURS-TRANS TO
             ESTIMATED-TASK-HOURS-MSTWORK
        MOVE ZEROS TO ACTUAL-TASK-HOURS-MSTWORK
        MOVE SPACES TO TASK-STATUS-INDICATORS-MSTWORK
        MOVE 'YES' TO WAS-IT-A-VALID-ADD-RECORD
     ELSE
        MOVE PROGRAM-IDENTIFICATION-TRANS TO
             PROGRAM-IDENTIFICATION-RPTWORK
        MOVE TRANSACTION-CODE-TRANS TO TRANSACTION-CODE-RPTWORK
        PERFORM C021-DETERMINE-ERROR-TYPE
        PERFORM U000-WRITE-EXCEPTION-REPORT.



 C021-DETERMINE-ERROR-TYPE.

     IF THE-TRANSACTION-IS-A-DELETION OR
           THE-TRANSACTION-IS-A-CHANGE
        MOVE NON-MATCHING-MASTER-ERRMSG TO TYPE-OF-ERROR-RPTWORK
     ELSE
        MOVE INVALID-TRANS-CODE-ERRMSG TO TYPE-OF-ERROR-RPTWORK.
****************************************************************
*                                                              *
*  THE FUNCTION OF THIS MODULE IS TO PROCESS A CHANGE          *
*  TRANSACTION. IT IS ENTERED FROM AND EXITS TO THE            *
*  C000-PROCESS-EQUAL-RECORDS MODULE.                          *
*                                                              *
****************************************************************

 D000-PROCESS-CHANGE-TRANS.

     IF PROGRAMMER-EMPL-NUMBER-TRANS NOT = SPACES
        MOVE PROGRAMMER-EMPL-NUMBER-TRANS TO
              PROGRAMMER-EMPL-NUMBER-OLDMST.
     PERFORM D001-CHECK-ACTUAL-HOURS
           VARYING ACTUAL-HOURS-ADD-SUBSCRIPT FROM 1 BY 1
           UNTIL ALL-ACTUAL-HOURS-ARE-ADDED.
     PERFORM D002-CHECK-STATUS-CODES
           VARYING STATUS-CODES-SUBSCRIPT FROM 1 BY 1
           UNTIL ALL-STATUS-CODES-ARE-CHECKED.



 D001-CHECK-ACTUAL-HOURS.

     IF SINGLE-ACTUAL-HOUR-A-TRANS (ACTUAL-HOURS-ADD-SUBSCRIPT)
           IS NOT = SPACES
        ADD SINGLE-ACTUAL-HOUR-TRANS (ACTUAL-HOURS-ADD-SUBSCRIPT)
              TO SINGLE-ACTUAL-HOUR-OLDMST
              (ACTUAL-HOURS-ADD-SUBSCRIPT).



 D002-CHECK-STATUS-CODES.

     IF SINGLE-STATUS-INDICATOR-TRANS (STATUS-CODES-SUBSCRIPT)
           = COMPLETED-CONSTANT
        MOVE COMPLETED-CONSTANT TO
              SINGLE-STATUS-INDICATOR-OLDMST
              (STATUS-CODES-SUBSCRIPT).
****************************************************************
*                                                              *
*  THE FUNCTION OF THIS UTILITY MODULE IS TO WRITE THE         *
*  EXCEPTION REPORT. IT IS ENTERED FROM AND EXITS TO SEVERAL   *
*  MODULES WITHIN THE PROGRAM (SEE HIERARCHY CHART).           *
*                                                              *
****************************************************************

 U000-WRITE-EXCEPTION-REPORT.

     IF THIS-IS-THE-FIRST-PAGE OR
           NUMBER-OF-LINES-PRINTED IS GREATER THAN PAGE-SIZE
        PERFORM U010-PRINT-HEADINGS.
     MOVE SPACES TO UPDATE-EXCEPTION-REPORT-LINE.
     MOVE PROGRAM-IDENTIFICATION-RPTWORK TO
           PROGRAM-IDENTIFICATION-REPORT.
     MOVE TRANSACTION-CODE-RPTWORK TO TRANSACTION-CODE-REPORT.
     MOVE TYPE-OF-ERROR-RPTWORK TO TYPE-OF-ERROR-REPORT.
     WRITE UPDATE-EXCEPTION-REPORT-LINE
           AFTER PROPER-SPACING.
     ADD PROPER-SPACING TO NUMBER-OF-LINES-PRINTED.
     MOVE SPACE-ONE-LINE TO PROPER-SPACING.
****************************************************************
*                                                              *
*  THIS MODULE PRINTS THE HEADINGS ON THE EXCEPTION REPORT. IT *
*  IS ENTERED FROM AND EXITS TO THE U000-WRITE-EXCEPTION-REPORT*
*  MODULE.                                                     *
*                                                              *
****************************************************************

 U010-PRINT-HEADINGS.

     MOVE CURRENT-DATE TO CURRENT-DATE-HDG1.
     MOVE PAGE-NUMBER-COUNT TO PAGE-NUMBER-HDG1.
     WRITE UPDATE-EXCEPTION-REPORT-LINE FROM FIRST-HEADING-LINE
           AFTER ADVANCING TO-THE-TOP-OF-THE-PAGE.
     ADD 1 TO PAGE-NUMBER-COUNT.
     WRITE UPDATE-EXCEPTION-REPORT-LINE FROM SECOND-HEADING-LINE
           AFTER ADVANCING 2 LINES.
     WRITE UPDATE-EXCEPTION-REPORT-LINE FROM THIRD-HEADING-LINE
           AFTER ADVANCING 1 LINES.
     MOVE ZERO TO NUMBER-OF-LINES-PRINTED.
     MOVE SPACE-TWO-LINES TO PROPER-SPACING.
```

BEST COPY

SEQUENTIAL FILE UPDATING

UNIT XI

ASSIGNMENT SHEET I: Introduction to Sequential File Updating

1. Name the four types of transactions used in sequential file updates and briefly explain the function of each.

	Name	Function
a.		
b.		
c.		
d.		

2. a. Which three types of transactions are used in this unit?

   b. Why does the sample program in this unit use only three types of transactions?

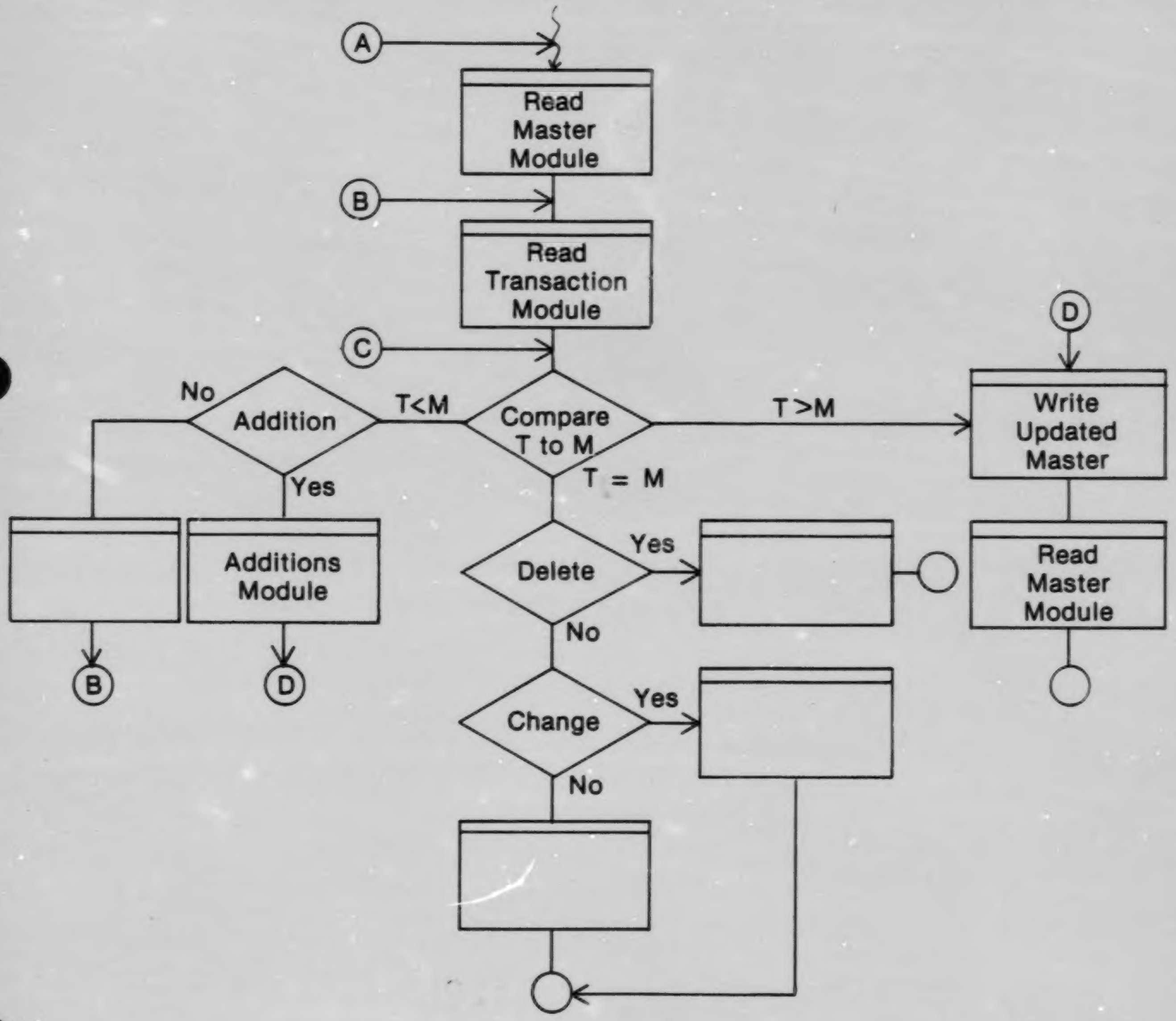
ASSIGNMENT SHEET 2: General Logic of Sequential File Update Programs

1. For each of the three logical conditions below, specify which of the conditions involve comparison of the transaction's ID to the master's ID. Identify the possible kind(s) of transactions which are valid and why.

Logical Condition	Valid Transaction
a. Transaction low	
b. Transaction equal	
c. Transaction greater than	

ASSIGNMENT SHEET 3: Flow chart of the General Logic of a Sequential Master File Update

1. Finish the flow chart below by filling in the correct major processing steps for each blank symbol.


A
Read Master Module
B
Read Transaction Module
C
D
No
Addition
T<M
Compare T to M
T>M
Write Updated Master
T = M
Yes
Additions Module
Delete
Yes
Read Master Module
B
D
No
Change
Yes
No

SEQUENTIAL FILE UPDATING

UNIT XI

ASSIGNMENT SHEET 4: Tracing the Flow of Records through Sequential File Update Logic

1. In the problem below, simulate the logic of processing a sequential file update program and finish the column on the right. Part of the answer is given. The input stream of transactions and master records is:

Transactions	Master
16 Add	15
17 chg	17
17 chg	18
18 delete	end file
end file	

	Processing Logic Condition		Processing, Output, and Read's
Comparison	Trans	Master	
(1)	16 Add:	15	a. Transaction greater than b. Write out master c. Read next master
(2)	16 Add:	17	a. Transaction less than b. Format new master c. Write out master d. Read next transaction
(3)	17 chg:	17	a. Transaction equal b. c.
(4)			
(5)			
(6)			
(7)			

SEQUENTIAL FILE UPDATING

UNIT XI

ANSWERS TO ASSIGNMENT SHEETS

Assignment Sheet 1

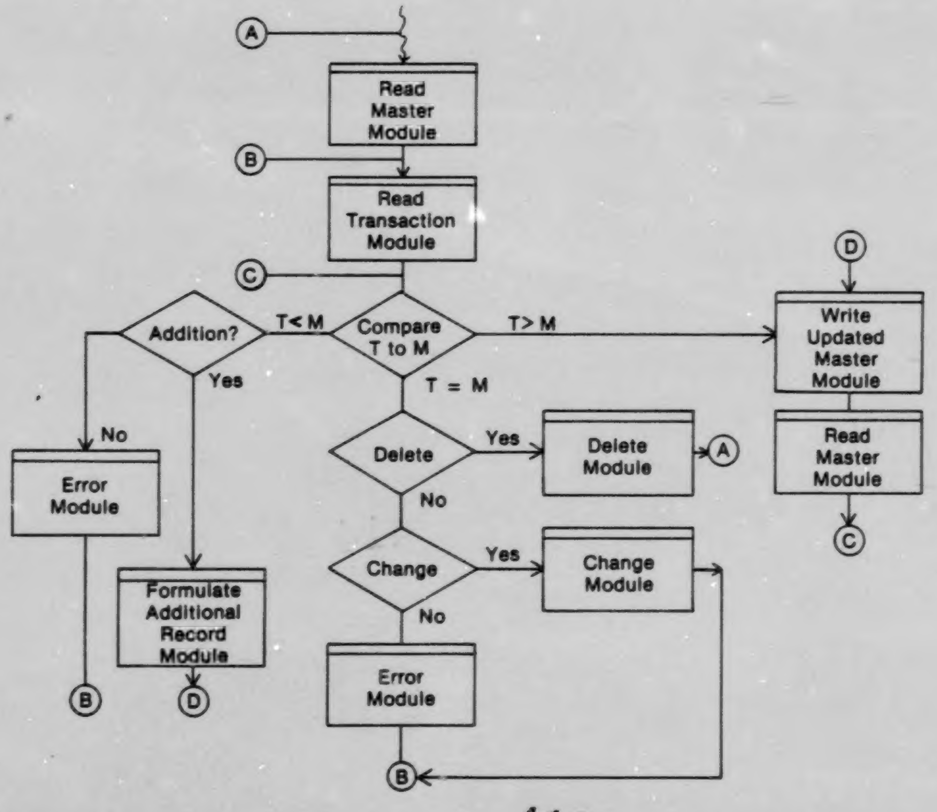
1. a. addition - adds a new master record
   b. deletion - removes an existing master record
   c. change - modifies values in an existing master record
   d. posting - applies financial results to master record values

2. a. addition, deletion, and change
   b. to simplify program logic

Assignment Sheet 2

1. a. addition - because this condition means that there is no existing record with this ID number
   b. changes and deletes - because this condition means that the ID's are equal and are there for the same entity
   c. no transaction and write master - because this condition means that all transactions for that particular ID are processed

Assignment Sheet 3


A
Read Master Module
B
Read Transaction Module
C
D
Addition?
T< M
Compare T to M
T> M
Write Updated Master Module
Yes
T = M
Read Master Module
No
Delete
Yes
Delete Module
A
Error Module
No
C
Change
Yes
Change Module
Formulate Additional Record Module
No
Error Module
B
D
B

Assignment Sheet 4

Processing Logic Condition			Processing, Output, and Read's
Comparison	Transaction	Master	
(1)	16 Add:	15	a. Transaction greater than b. Write out master c. Read next master
(2)	16 Add:	17	a. Transaction less than b. Format new master c. Write out master
(3)	17 chg:	17	a. Transaction equal b. Make changes c. Read next transaction
(4)	17 chg:	17	a. Transaction equal b. Make changes c. Read next transaction
(5)	18 delete:	17	a. Transaction greater than b. Write current master c. Read next master
(6)	18 delete:	18	a. Transaction equal b. Delete master c. Read next transaction d. Read next master
(7)	end file:	end file	a. End processing

SEQUENTIAL FILE UPDATING

UNIT XI

JOB SHEET 1: Writing a Complete Sequential File Update for Payroll (With an Appendix)

Materials

This job is to be done in COBOL on your school's computer. As part of the assignment, you will complete IPO charts and COBOL coding. Coding forms may be used if available. Otherwise, use clean, lined notepaper.

Introduction

A simple but complete sequential file update for payroll is the application for this job. In addition, you are provided with a similar sample application of a sequential file update program for a bibliography. The sample program will help you when designing and completing your payroll program, so study it carefully. At this time, turn to the appendix and carefully review the first four modules of the sample solved program.

Overview of the Payroll Application

Figure JS 1-1 is an overview of the job. In Figure JS 1-1, notice that three programs (or program segments) are depicted. These are:

1. H1 - create old master file
2. H2 - update payroll master
3. H3 - print payroll master

These three programs will later be combined into one actual program; but for purposes of easing the assignment, they have been divided into three segments. H1 reads the old master cards data (provided by your instructor as either cards or card images on disk) and creates the old payroll master file by writing each card into the output file and concurrently displaying the old master file information on the printer.

H2 is the center of the file update activity. It reads transactions (from cards or disk as provided by your instructor) and matches the transactions against the old master file to produce the new master file and a report of the transactions. H3 simply reads the new master file and formulates the records into a payroll master record report.

Starting the Job - H1

You will start the job by writing the procedure division for H1. Note that the work is considerably lessened because you will write only the procedure division for this application. The other three divisions will be provided by your instructor. Figure JS 1-2, Prewritten Divisions for Payroll Applications, is a printed copy of these first three divisions to be used in writing your procedure divisions.

Pseudocode Specification - H1

The pseudocode specifications for writing program H1 are given in Figure JS 1-3, Pseudocode Specification for Program H1. Input files are given in Figure JS 1-4. Note particularly that the old master file is opened for output in this module instead of being opened for input. This is because you will write the cards or card images into it to create the old master file and then close it. H2 will reopen it and specify that it will then be an input file. You will run H1 as part of H2 every time H2 is to be tested so that the old master file will exist for testing H2.

Completing H1

To complete H1, you should:

1. Draw IPO charts for each of the three modules.
2. Code each module in COBOL.
3. Test the program (see Figure JS 1-5).
4. Check the program output against Figure JS 1-6, Output of H1.
5. Hand in:
    a. IPO charts
    b. Coding sheets
    c. Successful COBOL compilation and execution printout

Starting the Job - H2

As in H1, you will only write the procedure division for program H2. You must successfully complete H1 before starting to test H2, as testing H2 requires the old master file from H1. At this time, review the remainder of the sample solved program in the appendix to this job sheet.

Figure JS 1-7, Hierarchy Chart for Payroll Sequential File Update Program, H2 has been provided for your use in writing the program. Carefully review it and begin drawing IPO charts for each module.

Figure JS 1-8, Pseudocode for Payroll Sequential File Update Program, H2 has also been provided to help you complete the program. You should complete the form by filling out the far right column in the form by reference to H1 and to the three prewritten divisions.

Note that the initialization module must reopen any files that were closed by the H1 modules. Also, the H1 modules are not shown in the H2 pseudocode because they are already written. A few standard routines such as NEW-PAGE, PAY-RTN, etc., are also not given; you should design these based on previous programming experience.

Completing - H2

To complete H2 you should:

1. Draw IPO charts for each of the three modules.
2. Write pseudocode for those modules not shown in Figure JS 1-8 (e.g, NEW-PAGE, PAY-RTN, etc.).
3. Test the program.
4. Check the program output against Figure JS 1-9.
5. Hand in:
    a. IPO charts
    b. Pseudocode
    c. Coding sheets
    d. Successful COBOL compilation and execution printout

Starting the Job - H3

The purpose of H3 is to print a master file content report based upon the outputted and updated new master payroll file from program H3. You should modify your stubbed modules in H2 to:

1. Open the new payroll master file for input.
2. Reset more records to yes.
3. Do a print routine until more records equal no.
4. Close the files.

Design of the hierarchy, IPO, pseudocode, and coding is left to you as this is a smaller assignment (generally 3 or 4 modules).

Completing H3

1. Draw hierarchy chart.
2. Draw IPO charts.
3. Write pseudocode for each module.
4. Test the program.
5. Check the program output against Figure JS 1-10. (Note that the output was not formatted as you would probably have done it.)
6. Hand in:

a. Hierarchy chart

b. IPO charts

c. Pseudocode

d. Coding sheets

e. Successful COBOL compilation and execution printout

Figure JS1-1: Payroll Application To Create, Update, and Print Payroll Master Files

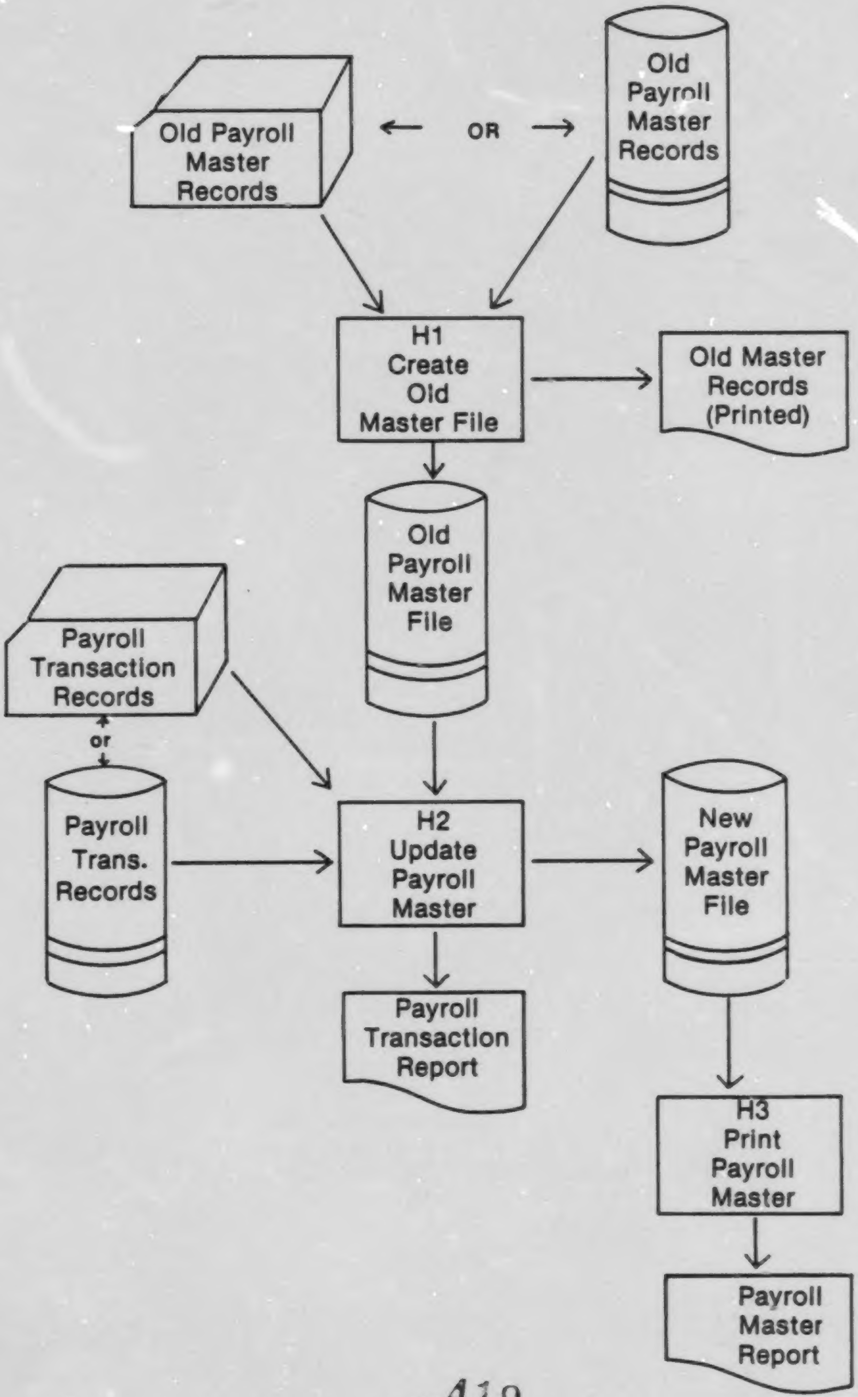

Old Payroll Master Records
OR
Old Payroll Master Records
H1 Create Old Master File
Old Master Records (Printed)
Old Payroll Master File
Payroll Transaction Records
or
Payroll Trans. Records
H2 Update Payroll Master
New Payroll Master File
Payroll Transaction Report
H3 Print Payroll Master
Payroll Master Report

Figure JS1-2: Prewritten Divisions for Payroll Applications, H2

```
 IDENTIFICATION DIVISION.
*
 PROGRAM-ID. PAYROL.
*
 ENVIRONMENT DIVISION.
*
 CONFIGURATION SECTION.
 SOURCE-COMPUTER. IBM-370.
 OBJECT-COMPUTER. IBM-370.
*
 SPECIAL-NAMES.
     C01 IS TOP-OF-PAGE.
*
 INPUT-OUTPUT SECTION.
 FILE-CONTROL.
     SELECT PRINT-FILE ASSIGN TO UR-S-SYSPRINT.
     SELECT E-TRANS-FILE ASSIGN TO UR-S-SYSIN.
     SELECT O-MAST-FILE ASSIGN TO UR-DA-S-SCRFILE1.
     SELECT N-MAST-FILE ASSIGN TO UR-DA-S-SCRFILE2.
*
 DATA DIVISION.
*
  FILE SECTION.
*
 FD  O-MAST-FILE
     LABEL RECORD IS OMITTED.
 01  O-MAST-REC                     PIC X(80).
*
 FD  N-MAST-FILE
     LABEL RECORD IS OMITTED.
 01  N-MAST-REC                     PIC X(80).
*
 FD  E-TRANS-FILE
     LABEL RECORD IS OMITTED.
 01  E-TRAN-REC                     PIC X(80).
*
 FD  PRINT-FILE
     LABEL RECORD IS OMITTED.
 01  PRINT-LINE                     PICTURE X(132).
*
 WORKING-STORAGE SECTION.
*
 77  LINE-COUNT                     PICTURE 99        VALUE 60.
 77  READ-SWITCH                    PICTURE X         VALUE 'B'.
*B=BOTH FILES,M=MASTER ONLY,T=TRANSACTION ONLY.
 77  ERROR-FLAG                     PICTURE X         VALUE 'N'.
*N=NO ERROR,Y=YES THERE IS AN ERROR.
 77  MASTER-STORE-SWITCH            PICTURE X         VALUE 'N'.
*N=NO MASTER STORED,Y=YES THERE IS A MASTER STORED.
 77  MORE-RECORDS                   PICTURE X         VALUE 'Y'.
*Y=YES THERE ARE  MORE RECORDS,N=NO MORE RECORDS.
*
 01  WS-MAST-REC.
     05  O-EMP-NAME                 PIC X(25).
     05  O-EMP-ID                   PIC X(10).
     05  O-DEPT                     PIC X(15).
     05  O-PAY-RATE                 PIC 99V99.
     05  O-GPAY-YTD                 PIC 99999V99.
     05  O-NPAY-YTD                 PIC 99999V99.
     05  FILLER                     PICTURE X(12).
*
 01  WS-TRAN-REC.
     05  E-EMP-NAME                 PIC X(25).
     05  E-EMP-ID                   PIC X(10).
     05  E-DEPT                     PIC X(15).
     05  E-PAY-RATE                 PIC 99V99.
     05  E-GPAY                     PIC 9999V99.
     05  E-NPAY                     PIC 9999V99.
     05  FILLER                     PICTURE X(11).
     05  T-CODE                     PIC XXX.
```

```
*
 01  REPORT-TITLE.
     05  FILLER                     PICTURE X(50)   VALUE SPACES.
     05  FILLER                     PICTURE X(33)   VALUE
             'PAYROLL UPDATE TRANSACTION REPORT'.
     05  FILLER                     PICTURE X(49)   VALUE SPACES.
*
 01  HEAD1.
     05  FILLER                     PICTURE X(4)    VALUE SPACES.
     05  FILLER                     PICTURE X(5)    VALUE 'TRANS'
     05  FILLER                     PICTURE X(72)   VALUE SPACES.
     05  FILLER                     PICTURE X(9)    VALUE
                            'RATE     '.
     05  FILLER                     PICTURE X(20)   VALUE
                            'GROSS PAY    NET PAY'.
     05  FILLER                     PICTURE X(22)   VALUE SPACES.
*
 01  HEAD2.
     05  FILLER                     PICTURE X(4)    VALUE SPACES.
     05  FILLER                     PICTURE X(4)    VALUE 'CODE'.
     05  FILLER                     PICTURE X(5)    VALUE SPACES.
     05  FILLER                     PICTURE X(15)   VALUE
                            'ID NUMBER      '.
     05  FILLER                     PICTURE X(30)   VALUE
                            'EMPLOYEE NAME                 '.
     05  FILLER                     PICTURE X(22)   VALUE
                            'DEPARTMENT            '.
     05  FILLER                     PICTURE X(30)   VALUE
                            '[illegible] PAY     THIS PD     THIS PD'.
     05  FILLER                     PICTURE X(22)   VALUE
                            '  CAUSE OF ERROR      '.
*
 01  D-DETAIL-LINE.
     05  FILLER                     PICTURE X(5)    VALUE SPACES.
     05  D-T-CODE                 PIC XXX.
     05  FILLER                     PICTURE X(5)    VALUE SPACES.
     05  D-EMP-ID                 PIC X(10).
     05  FILLER                     PICTURE X(5)    VALUE SPACES.
     05  D-EMP-NAME               PIC X(25).
     05  FILLER                     PICTURE X(5)    VALUE SPACES.
     05  D-DEPT                   PIC X(15).
     05  FILLER                     PICTURE X(7)    VALUE SPACES.
     05  D-PAY-RATE               PIC $$9.99.
     05  FILLER                     PICTURE X(5)    VALUE SPACES.
     05  D-GPAY                   PIC $$$$.99.
     05  FILLER                     PICTURE X(5)    VALUE SPACES.
     05  D-NPAY                   PIC  $$$$.99.
     05  FILLER                     PICTURE X(2)    VALUE SPACES.
     05  CAUSE-OF-ERROR             PICTURE X(20).
*

 01  TOTAL-LINE.
     05  FILLER                     PICTURE X(10)   VALUE SPACES.
     05  FILLER                     PICTURE X(19)   VALUE
                            'TOTAL TRANSACTIONS:'.
     05  TOT-TRANS              PIC ZZZZ9.
     05  FILLER                     PICTURE X(15)   VALUE
                            '       CORRECT:'.
     05  TOT-RIGHT              PIC ZZZZ9.
     05  FILLER                     PICTURE X(14)   VALUE
                            '       ERRORS:'.
     05  TOT-WRONG              PIC ZZZZ9.
     05  FILLER                     PICTURE X(23)   VALUE
                            '       TOTAL GROSS PAY:'.
     05  TOT-GPAY               PIC $$$$$9.99.
     05  FILLER                     PICTURE X(10)   VALUE
                            '  NET PAY:'.
     05  TOT-NPAY               PIC $$$$$9.99.
     05  FILLER                     PICTURE X(6)    VALUE SPACES.
```

Instructions: Use these divisions to obtain data, file record names, etc..for the PROCEDURE DIVISION of your own program.

Figure JS1-2: Prewritten Divisions for Payroll Applications (cont'd)

```
*
 01  T-TOTALS.
     05  T-TOT-TRANS            PIC 9999        VALUE ZERO.
     05  T-TOT-RIGHT            PIC  9999       VALUE ZERO.
     05  T-TOT-WRONG            PIC 9999        VALUE ZERO.
     05  T-TOT-GPAY             PIC 9999999V99  VALUE ZERO.
     05 T-TOT-NPAY              PIC 9999999V99  VALUE ZERO.
*
 01  TEMP-MAST-STORE            PIC X(84).
*
*
 01  S-NMASTR-TRAILER.
     05  FILLER                    PIC X(25).
     05  S-EMP-ID                    PIC X(10) VALUE IS
                                   'ZZZ-999999'.
     05  FILLER                    PIC X(19).
     05  S-GP                      PIC 9(5)V99 VALUE IS ZERO.
     05  S-NP                      PIC 9(5)V99 VALUE IS ZERO.
     05  S-NR                      PIC 9(5)V99 VALUE IS ZERO.
     05  FILLER                    PIC X(5).
*
 01  C-TRAN-TOTALS.
     05  C-EMP-ID                    PIC X(10) VALUE IS
                                   'ZZZ-999999'.
     05  C-GP                      PIC 9(5)V99 VALUE IS ZERO.
     05  C-NP                      PIC 9(5)V99 VALUE IS ZERO.
     05  C-NR                      PIC 9(5)V99 VALUE IS ZERO.
*
 01  F-OMAST-TOTALS.
     05  F-EMP-ID                    PIC X(10) VALUE IS
                                   'ZZZ-999999'.
     05  F-GP                      PIC 9(5)V99 VALUE IS ZERO.
     05  F-NP                      PIC 9(5)V99 VALUE IS ZERO.
     05  F-NR                      PIC 9(5)V99 VALUE IS ZERO.
*
*
*
 IDENTIFICATION DIVISION.
 PROGRAM-ID. PAYROL.
*
 ENVIRONMENT DIVISION.
*
 CONFIGURATION SECTION.
 SOURCE-COMPUTER. IBM-370.
 OBJECT-COMPUTER. IBM-370.
*
 SPECIAL-NAMES.
     C01 IS TOP-OF-PAGE.
*
 INPUT-OUTPUT SECTION.
 FILE-CONTROL.
     SELECT PRINT-FILE ASSIGN TO UR-S-SYSPRINT.
     SELECT E-TRANS-FILE ASSIGN TO UR-S-SYSIN.
     SELECT O-MAST-FILE ASSIGN TO UR-DA-S-UMCSCR1.
     SELECT N-MAST-FILE ASSIGN TO UR-DA-S-UMCSCR2.
*
 DATA DIVISION.
*
  FILE SECTION.
*
 FD  O-MAST-FILE
     LABEL RECORD IS OMITTED.
 01  O-MAST-REC                    PIC X(80).
*
 FD  N-MAST-FILE
     LABEL RECORD IS OMITTED.
 01  N-MAST-REC.
     05  FILLER                   PICTURE X(25).
     05  N-EMP-ID                PIC X(10).
     05  FILLER                   PICTURE X(45).
*
 FD  E-TRANS-FILE
     LABEL RECORD IS OMITTED.
 01  E-TRAN-REC                        PIC X(80).
*
 FD  PRINT-FILE
     LABEL RECORD IS OMITTED.
 01  PRINT-LINE                   PICTURE X(132).
*
 WORKING-STORAGE SECTION.
```

```
*
 77  LINE-COUNT                 PICTURE 99      VALUE 60.
 77  READ-SWITCH                PICTURE X        VALUE 'B'.
*B=BOTH FILES,M=MASTER ONLY,T=TRANSACTION ONLY.
 77  ERROR-FLAG                 PICTURE X        VALUE 'N'.
*N=NO ERROR,Y=YES THERE IS AN ERROR.
 77  MASTER-STORE-SWITCH        PICTURE X        VALUE 'N'.
*N=NO MASTER STORED,Y=YES THERE IS A MASTER STORED.
 77  MORE-RECORDS               PICTURE X        VALUE 'Y'.

*Y=YES THERE ARE  MORE RECORDS,N=NO MORE RECORDS.
*
 01  WS-MAST-REC.
     05  O-EMP-NAME              PIC X(25).
     05  O-EMP-ID                PIC X(10).
     05  O-DEPT                  PIC X(15).
     05  O-PAY-RATE             PIC 99V99.
     05  O-GPAY-YTD              PIC 99999V99.
     05  O-NPAY-YTD              PIC 99999V99.
     05  FILLER                   PICTURE X(12).

*
 01  WS-TRAN-REC.
     05  E-EMP-NAME              PIC X(25).
     05  E-EMP-ID                PIC X(10).
     05  E-DEPT                  PIC X(15).
     05  E-PAY-RATE              PIC 99V99.
     05  E-GPAY                PIC 9999V99.
     05  E-NPAY                  PIC 9999V99.
     05  FILLER                   PICTURE X(11).
     05  T-CODE                PIC XXX.
*
 01  REPORT-TITLE.
     05  FILLER                   PICTURE X(50)   VALUE SPACES.
     05  FILLER                   PICTURE X(33)   VALUE
          'PAYROLL UPDATE TRANSACTION REPORT'.
     05  FILLER                   PICTURE X(49)   VALUE SPACES.
*
 01  HEAD1.
     05  FILLER                   PICTURE X(4)    VALUE SPACES.
     05  FILLER                   PICTURE X(5)    VALUE 'TRANS'.
     05  FILLER                   PICTURE X(72)   VALUE SPACES.
     05  FILLER                   PICTURE X(9)    VALUE
                          'RATE      '.
     05  FILLER                   PICTURE X(20)   VALUE
                          'GROSS PAY    NET PAY'.
     05  FILLER                   PICTURE X(22)   VALUE SPACES.
*
 01  HEAD2.
     05  FILLER                   PICTURE X(4)    VALUE SPACES.
     05  FILLER                   PICTURE X(4)    VALUE 'CODE'.
     05  FILLER                   PICTURE X(5)    VALUE SPACES.
     05  FILLER                   PICTURE X(15)   VALUE
                          'ID NUMBER     '.
     05  FILLER                   PICTURE X(30)   VALUE
                          'EMPLOYEE NAME                '.
     05  FILLER                   PICTURE X(22)   VALUE
                          'DEPARTMENT            '.
     05  FILLER                   PICTURE X(30)   VALUE
                          'OF PAY      THIS PD      THIS PD'.
     05  FILLER                   PICTURE X(22)   VALUE
                          '  CAUSE OF ERROR       '.
*
 01  D-DETAIL-LINE.
     05  FILLER                   PICTURE X(5)    VALUE SPACES.
     05  D-T-CODE               PIC XXX.

     05  FILLER                   PICTURE X(5)    VALUE SPACES.
     05  D-EMP-ID               PIC X(10).
     05  FILLER                   PICTURE X(5)    VALUE SPACES.
     05  D-EMP-NAME             PIC X(25).
     05  FILLER                   PICTURE X(5)    VALUE SPACES.
     05  D-DEPT                 PIC X(15).
     05  FILLER                   PICTURE X(7)    VALUE SPACES.
     05  D-PAY-RATE             PIC $$9.99.
     05  FILLER                   PICTURE X(5)    VALUE SPACES.
     05  D-GPAY                 PIC $$$$.99.
     05  FILLER                   PICTURE X(5)    VALUE SPACES.
     05  D-NPAY                 PIC  $$$$.99.
     05  FILLER                   PICTURE X(2)    VALUE SPACES.
     05  CAUSE-OF-ERROR            PICTURE X(20).
```

Figure JS1-2: Prewritten Divisions for Payroll Applications (cont'd)

```
*
 01  TEMP-MAST-STORE        PIC X(84).
*
*
 01  S-NMASTR-TRAILER.
     05  FILLER                   PIC X(25).
     05  S-EMP-ID                   PIC X(10) VALUE IS
                                  'ZZZ-999999'.
     05  FILLER                   PIC X(19).
     05  S-GP                     PIC 9(5)V99 VALUE IS ZERO.
     05  S-NP                     PIC 9(5)V99 VALUE IS ZERO.
     05  S-NR                     PIC 9(5)V99 VALUE IS ZERO.
     05  FILLER                   PIC X(5).

*
 01  TOTAL-LINE.
     05  FILLER                 PICTURE X(10)   VALUE SPACES.
     05  FILLER                 PICTURE X(19)   VALUE
                          'TOTAL TRANSACTIONS:'.
     05  TOT-TRANS          PIC ZZZZ9.
     05  FILLER                 PICTURE X(15)   VALUE
                          '         CORRECT:'.
     05  TOT-RIGHT          PIC ZZZZ9.
     05  FILLER                 PICTURE X(14)   VALUE
                          '         ERRORS:'.
     05  TOT-WRONG          PIC ZZZZ9.
     05  FILLER                 PICTURE X(23)   VALUE
                          '         TOTAL GROSS PAY:'.
     05  TOT-GPAY           PIC $$$$$9.99.
     05  FILLER                 PICTURE X(10)   VALUE
                          '  NET PAY:'.
     05  TOT-NPAY           PIC $$$$$9.99.
     05  FILLER                 PICTURE X(6)    VALUE SPACES.
*
 01  T-TOTALS.
     05  T-TOT-TRANS            PIC 9999        VALUE ZERO.
     05  T-TOT-RIGHT           PIC  9999       VALUE ZERO.
     05  T-TOT-WRONG            PIC 9999        VALUE ZERO.
     05  T-TOT-GPAY             PIC 9999999V99  VALUE ZERO.
      05 T-TOT-NPAY             PIC 9999999V99  VALUE ZERO.

 *
  01  C-TRAN-TOTALS.

     05  C-EMP-ID                   PIC X(10) VALUE IS
                                  'ZZZ-999999'.
     05  C-GP                     PIC 9(5)V99 VALUE IS ZERO.
     05  C-NP                     PIC 9(5)V99 VALUE IS ZERO.
     05  C-NR                     PIC 9(5)V99 VALUE IS ZERO.
*
 01  F-OMAST-TOTALS.
     05  F-EMP-ID                   PIC X(10) VALUE IS
                                  'ZZZ-999999'.
     05  F-GP                     PIC 9(5)V99 VALUE IS ZERO.
     05  F-NP                     PIC 9(5)V99 VALUE IS ZERO.
     05  F-NR                     PIC 9(5)V99 VALUE IS ZERO.
```

Figure JS1-3: Pseudocode Specifications for Program H1

Pseudocode Specifications

DATE: May 2	**PROGRAMMER:** Jane Austin	**PROGRAM:** H1
MODULE FUNCTION: Create Old Master	**REF:**	**MODULE NAME:** Create Old Mast

```
PSEUDOCODE                                        REF:   FILES, RECORDS, FIELDS
                                                         REQUIRED

MAIN-CONTROL-RTN-MODULE
  Do initialization
  Stop run

INITIALIZATION MODULE                                    O-MAST-FILE, E-TRANS-FILE
  Open files (old mast for output)
  Do mast-create-rtn
    until more-recs = no
  Close old mast, tran-file

MASTER-CREATE-RTN MODULE                                 E-TRANS-FILE, WS-MAST-REC
  Read tran-file into working storage
        master record
        at end display error.      { note: ID is
  If mast-id = 999 etc.            { 'zzz-999999'
      move no to more-rec.         { at end of file
  Display ws-mast.
  Write old-mast from ws-mast                            WS-MAST-REC, O-MAST-REC
```

Figure JS1-4: Input Files for use in H1, H2, and H3 COBOL Programs

Part A: Old master records for creating old master payroll file

```
BUTLER, RHETT            BUT-824571MARKETING      087503500000262500
FAVIR, LARRY             FAV-835127MARKETING      070002800000210000
FINKBINE, GEORGE         FIN-571433ACCOUNTING     047501900000150000
HAMILTON, MELANIE        HAM-446735DATA PROCESSING038501732500138600
HAMPTON, WADE            HAM-642151DATA PROCESSING043201728000138240
IRWIN, MONTAGUE          IRW-924586ACCOUNTING     085603424000256800
MACINTOSH, ANGUS         MAC-662473MARKETING      053002226000178080
MASTERN, CASS            MAS-619318MARKETING      045001912500153000
MERRIWETHER, MAYBELLE    MER-527348DATA PROCESSING027501100000093500
OHARA SCARLETT           OHA-431552MARKETING      045001800000144000
PORTNOY, ALEXANDER       POR-783914DATA PROCESSING100004000000300000
ROBILLARD, ELLEN         ROB-445331DATA PROCESSING036001440000115200
SLATTERY, THOMAS         SLA-108423ACCOUNTING     093503740000280500
STANTON, ANNE            STA-274432DATA PROCESSING090503740000280500
STARK, WILLIE            STA-367810MARKETING      060202528000202240
TARLETON, BRENT          TAR-311834DATA PROCESSING037501500000120000
WATLING, BELLE           WAT-623489MARKETING      063202528000189600
WYNDER, ABLE             WYN-246372ACCOUNTING     075003000000240000
                         ZZZ-999999                   4911900379966020
```

Part B: Transaction records for input into payroll update program, H2

```
                         BUR-145927                   015800012640          PAY
BURDEN, JOHN             BUR-382196                                         CHG
CALVERT, CATHLEEN        CAL-336741MARKETING      0625                      ADD
                         FAV-835127                   019000015200          DEL
                         FIN-571433                   019000015200          PAY
                         HAM-446735ACCOUNTING                               CHG
IRWIN, MONTAGUE          IRW-924586ACCOUNTING     085603424000256800        DEL
                         MAS-619318PRODUCTION     0500                      CHG
                         MAS-619318                   020000016000          PAY
                         ROB-445331               049903180002540 0         CHG
                         SLA-108423                   037400028050          PAY
WILKES, ASHLEY           WIL-300177ACCOUNTING     0835                      ADD
ZIPPER,ANN               ZIP-714000CLERICAL       045000000000000000        ADD
                         ZZZ-999999                     92200  71890  12
```

Figure JS1-5: Steps for Testing H1

1. Get JCL deck from instructor.
2. Duplicate JCL cards.
3. Return JCL deck to instructor.
4. Put cards in following order:

```
// Jobcard
//          PASSWORD=,
//          MSGLEVEL=(1,1)
//STEP EXEC COBUCG,TIME.GO=(,2)
//COB.SYSIN DD
DSN=$1136TK.PUBLIC.MASTER.FILE(IDDIV),DISP=SHR
//          DD  *

            YOUR PROCEDURE DIVISION

//GO.SCRFILE1 DD UNIT=SYSVIO,SPACE=(TRK,(2,1),RLSE)
//GO.SCRFILE2 DD UNIT=SYSVIO,SPACE=(TRK,(2,1),RLSE)
//GO.SYSIN DD
DSN=$1136TK.PUBLIC.MASTER.FILE(DATA),DISP=SHR
//
```

5. Run program.
6. Check output with Figure JS1-6.

Note: This represents sample steps. It may be different at your school.

Figure JS1-6: Output of H1

```
BURKE, SADIE             BUR-145927DATA PROCESSING039501580000126400
BURDEN, JACK             BUR-382196ACCOUNTING     073502940000220500
BUTLER, RHETT            BUT-824571MARKETING      087503500000262500
FAVIR, LARRY             FAV-835127MARKETING      070002800000210000
FINKBINE, GEORGE         FIN-571433ACCOUNTING     047501900000150000
HAMILTON, MELANIE        HAM-446735DATA PROCESSING038501732500138600
HAMPTON, WADE            HAM-642151DATA PROCESSING043201728000138240
IRWIN, MONTAGUE          IRW-924586ACCOUNTING     085603424000256800
MACINTOSH, ANGUS         MAC-662473MARKETING      053002226000178080
MASTERN, CASS            MAS-619318MARKETING      045001912500153000
MERRIWETHER, MAYBELLE    MER-527348DATA PROCESSING027501100000093500
MORRISON, JAMES          MOR-754884DATA PROCESSING057301942000156700
OHARA SCARLETT           OHA-431552MARKETING      045001800000144000
PORTNOY, ALEXANDER       POR-783914DATA PROCESSING100004000000300000
ROBILLARD, ELLEN         ROB-445331DATA PROCESSING036001440000115200
SLATTERY, THOMAS         SLA-108423ACCOUNTING     093503740000280500
STANTON, ANNE            STA-274432DATA PROCESSING090503740000280500
STARK, WILLIE            STA-367810MARKETING      060202528000202240
TARLETON, BRENT          TAR-311834DATA PROCESSING037501500000120000
WATLING, BELLE           WAT-623489MARKETING      063202528000189600
WYNDER, ABLE             WYN-246372ACCOUNTING     075003000000240000
                         ZZZ-999999                   4911900379966020
```

Figure JS1-7: Hierarchy Chart for Payroll Sequential File Update Program, H2

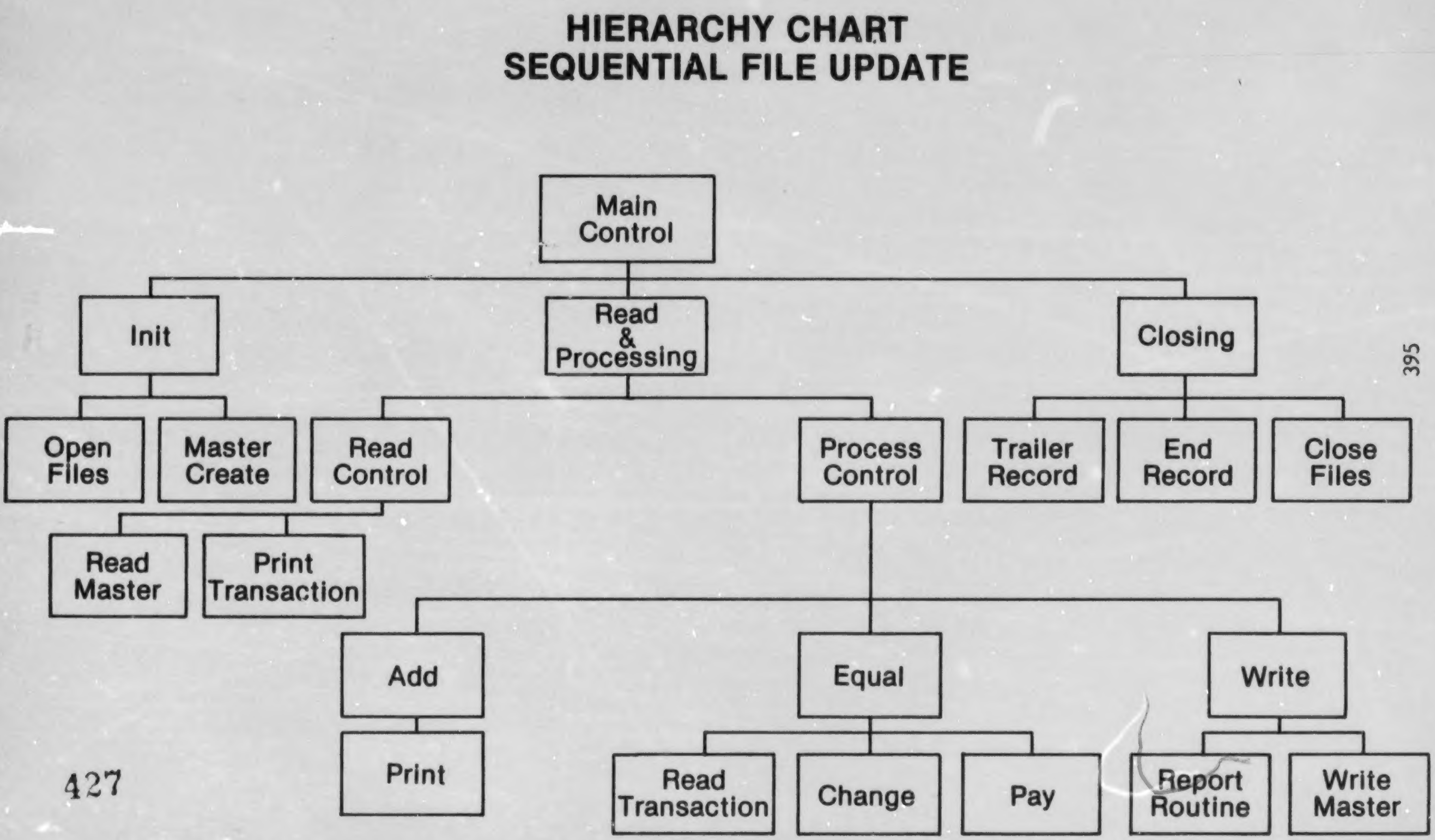
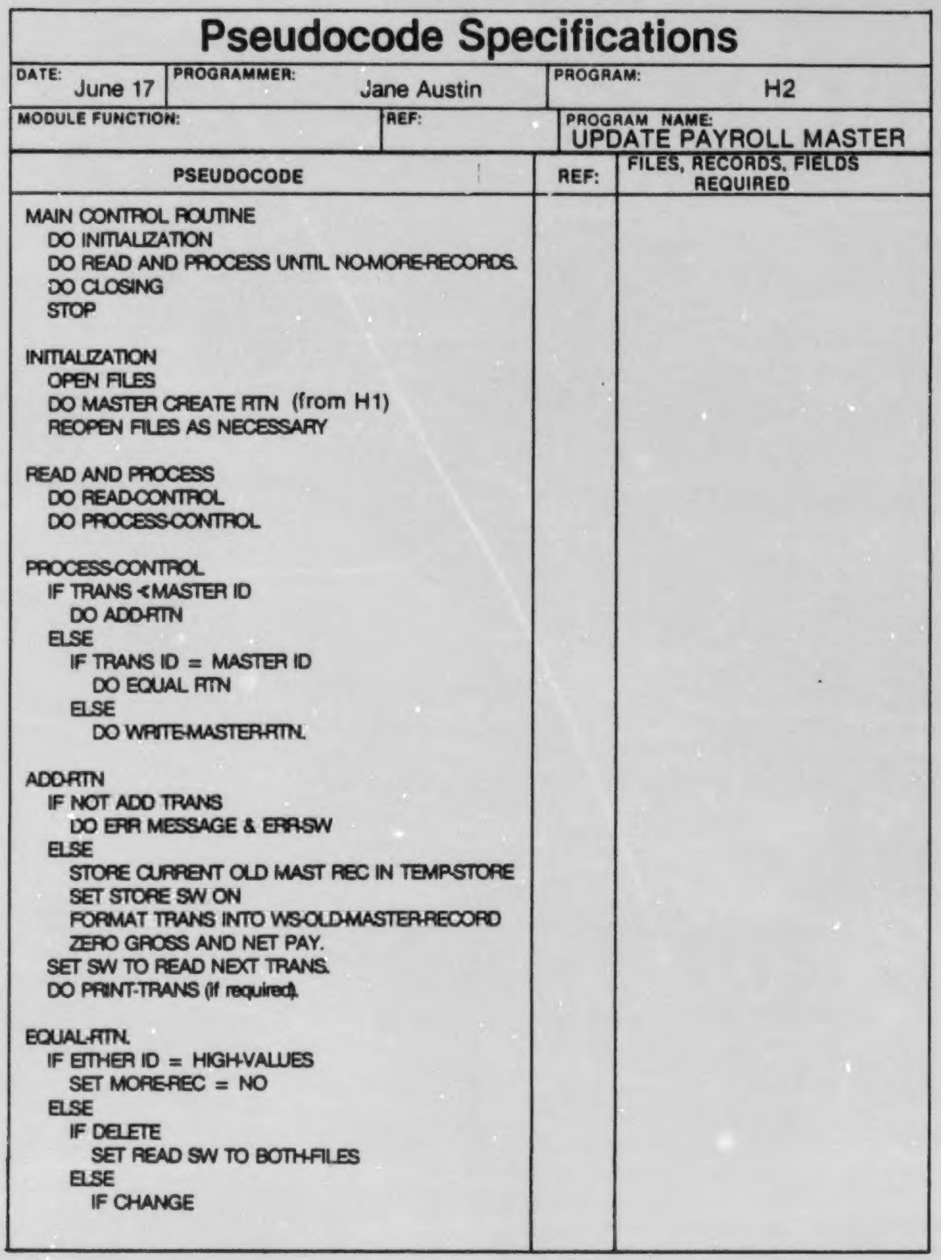

HIERARCHY CHART
SEQUENTIAL FILE UPDATE
Main Control
Init
Read & Processing
Closing
Open Files
Master Create
Read Control
Process Control
Trailer Record
End Record
Close Files
Read Master
Print Transaction
Add
Equal
Write
Print
Read Transaction
Change
Pay
Report Routine
Write Master

Figure JS1-8: Pseudocode for Payroll Sequential File Update Program, H2
(Part 1 of 2)

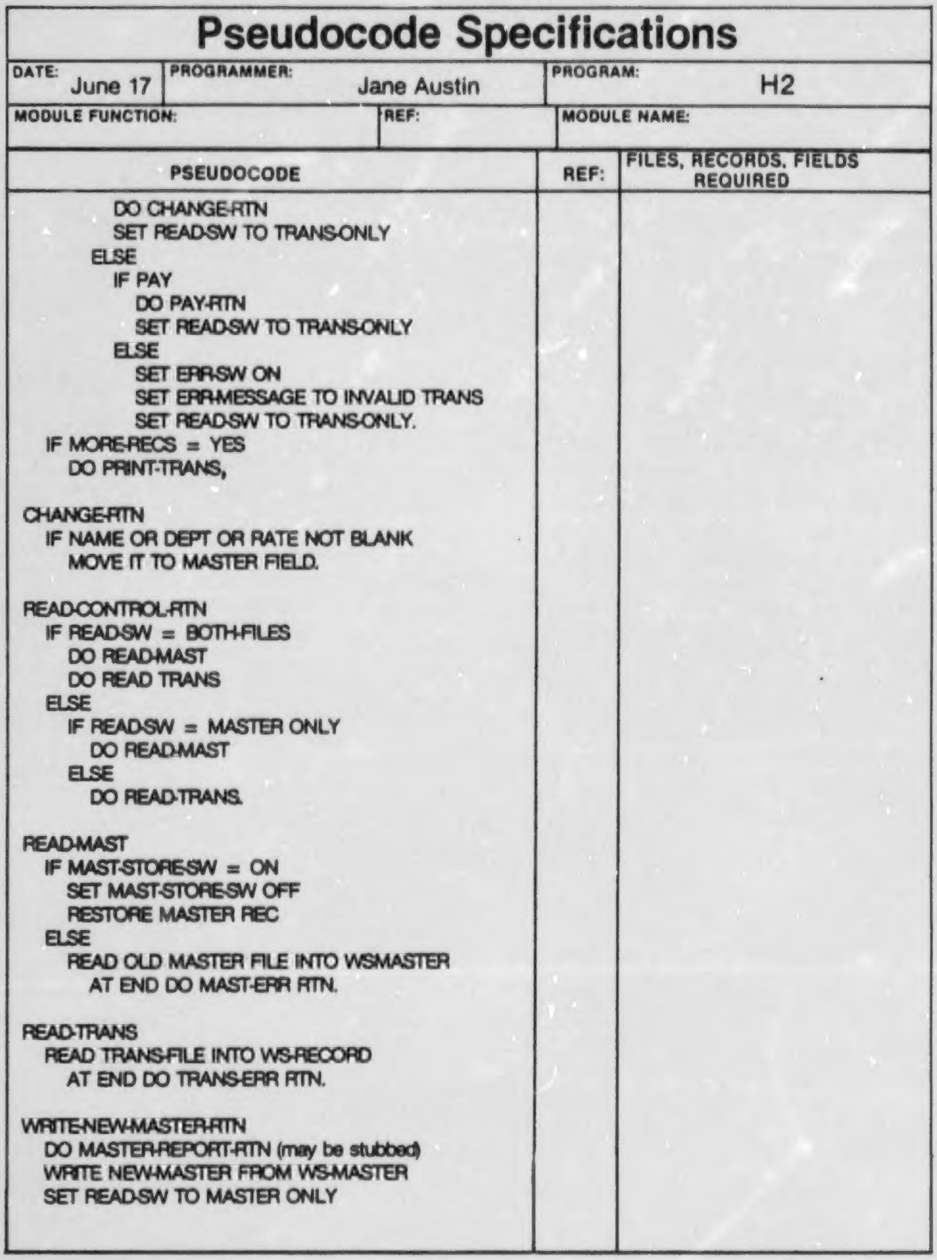


Pseudocode Specifications

DATE: June 17	PROGRAMMER: Jane Austin	PROGRAM: H2
MODULE FUNCTION:	REF:	PROGRAM NAME: UPDATE PAYROLL MASTER
PSEUDOCODE	REF:	FILES, RECORDS, FIELDS REQUIRED

```
MAIN CONTROL ROUTINE
  DO INITIALIZATION
  DO READ AND PROCESS UNTIL NO-MORE-RECORDS.
  DO CLOSING
  STOP

INITIALIZATION
  OPEN FILES
  DO MASTER CREATE RTN (from H1)
  REOPEN FILES AS NECESSARY

READ AND PROCESS
  DO READ-CONTROL
  DO PROCESS-CONTROL.

PROCESS-CONTROL
  IF TRANS <MASTER ID
    DO ADD-RTN
  ELSE
    IF TRANS ID = MASTER ID
      DO EQUAL RTN
    ELSE
      DO WRITE-MASTER-RTN.

ADD-RTN
  IF NOT ADD TRANS
    DO ERR MESSAGE & ERR-SW
  ELSE
    STORE CURRENT OLD MAST REC IN TEMP-STORE
    SET STORE SW ON
    FORMAT TRANS INTO WS-OLD-MASTER-RECORD
    ZERO GROSS AND NET PAY.
  SET SW TO READ NEXT TRANS.
  DO PRINT-TRANS (if required).

EQUAL-RTN.
  IF EITHER ID = HIGH-VALUES
    SET MORE-REC = NO
  ELSE
    IF DELETE
      SET READ SW TO BOTH-FILES
    ELSE
      IF CHANGE
```

Figure JS1-8: Pseudocode for Payroll Sequential File Update Program, H2
(Part 2 of 2)



Pseudocode Specifications

DATE: June 17	PROGRAMMER: Jane Austin	PROGRAM: H2
MODULE FUNCTION:	REF:	MODULE NAME:

PSEUDOCODE	REF:	FILES, RECORDS, FIELDS REQUIRED

```
                DO CHANGE-RTN
                SET READ-SW TO TRANS-ONLY
            ELSE
                IF PAY
                    DO PAY-RTN
                    SET READ-SW TO TRANS-ONLY
                ELSE
                    SET ERR-SW ON
                    SET ERR-MESSAGE TO INVALID TRANS
                    SET READ-SW TO TRANS-ONLY.
    IF MORE-RECS = YES
        DO PRINT-TRANS,

CHANGE-RTN
    IF NAME OR DEPT OR RATE NOT BLANK
        MOVE IT TO MASTER FIELD.

READ-CONTROL-RTN
    IF READ-SW = BOTH-FILES
        DO READ-MAST
        DO READ TRANS
    ELSE
        IF READ-SW = MASTER ONLY
            DO READ-MAST
        ELSE
            DO READ-TRANS.

READ-MAST
    IF MAST-STORE-SW = ON
        SET MAST-STORE-SW OFF
        RESTORE MASTER REC
    ELSE
        READ OLD MASTER FILE INTO WS-MASTER
            AT END DO MAST-ERR RTN.

READ-TRANS
    READ TRANS-FILE INTO WS-RECORD
        AT END DO TRANS-ERR RTN.

WRITE-NEW-MASTER-RTN
    DO MASTER-REPORT-RTN (may be stubbed)
    WRITE NEW-MASTER FROM WS-MASTER
    SET READ-SW TO MASTER ONLY
```

Figure JS1-9: Output from Program H2

```
                                               PAYROLL UPDATE TRANSACTION REPORT
TRANS                                                                          RATE      GROSS PAY      NET PAY
CODE     ID NUMBER      EMPLOYEE NAME               DEPARTMENT                OF PAY      THIS PD       THIS PD   CAUSE OF ERROR

 PAY     BUR-145927                                                           $0.00      $158.00       $126.40

 CHG     BUR-382196     BURDEN, JOHN                                          $0.00         $.00          $.00

 ADD     CAL-336741     CALVERT, CATHLEEN           MARKETING                 $6.25         $.00          $.00

 DEL     FAV-835127                                                           $0.00      $190.00       $152.00

 PAY     FIN-571433                                                           $0.00      $190.00       $152.00

 CHG     HAM-446735                                 ACCOUNTING                $0.00         $.00          $.00

 DEL     IRW-924586     IRWIN, MUNTAGUE             ACCOUNTING                $8.56      $342.40        $25.68

 CHG     MAS-619318                                 PRODUCTION                $5.00         $.00          $.00

 PAY     MAS-619318                                                           $0.00      $200.00       $160.00

 CHG     ROB-445331                                                           $4.99      $318.00       $254.00

 PAY     SLA-108423                                                           $0.00      $374.00       $280.50

 PAY     STA-274432                                                           $9.05      $400.50       $302.75

 ADD     SUT-654321     SUTTER, BRUCE               DATA PROCESSING          $15.00         $.00          $.00

 ADD     WIL-300177     WILKES, ASHLEY              ACCOUNTING                $8.35         $.00          $.00

 ADD     ZIP-714000     ZIPPER, ANN                 CLERICAL                  $4.50         $.00          $.00

 TOTAL TRANSACTIONS:    15       CORRECT:    15   ERRORS:    0     TOTAL GROSS PAY:    $1322.50     NET PAY:    $1021.65
```

Figure JS1-10: Output from Program H3

```
BURKE, SADIE                     BUR-145927DATA PROCESSING039501738000139040
BURDEN, JOHN                     BUR-382196ACCOUNTING     073502940000220500
BUTLER, RHETT                    BUT-824571MARKETING      087503500000262500
CALVERT, CATHLEEN                CAL-336741MARKETING      06250     00     0
FINKBINE, GEORGE                 FIN-571433ACCOUNTING     047502090000165200
HAMILTON, MELANIE                HAM-446735DATA PROCESSING038501732500138600
HAMPTON, WADE                    HAM-642151DATA PROCESSING043201728000138240
MACINTOSH, ANGUS                 MAC-662473MARKETING      053002226000178080
MASTERN, CASS                    MAS-619318MARKETING      050002112500169000
MERRIWETHER, MAYBELLE            MER-527348DATA PROCESSING027501100000093500
MORRISON, JAMES                  MOR-754884DATA PROCESSING057301942000156700
OHARA, SCARLETT                  OHA-431552MARKETING      045001800000144000
PORTNOY, ALEXANDER               POR-783914DATA PROCESSING100004000000300000
ROBILLARD, ELLEN                 ROB-445331DATA PROCESSING049901440000115200
SLATTERY, THOMAS                 SLA-108423ACCOUNTING     093504114000308550
STANTON, ANNE                    STA-274432DATA PROCESSING090504140500310775
STARK, WILLIE                    STA-367810MARKETING      060202528000202240
SUTTER, BRUCE                    SUT-654321DATA PROCESSING15000     00     0
TARLETON, BRENT                  TAR-311834DATA PROCESSING037501500000120000
WATLING, BELLE                   WAT-623489MARKETING      063202528000189600
WILKES, ASHLEY                   WIL-300177ACCOUNTING     08350     00     0
WYNDER, ABLE                     WYN-246372ACCOUNTING     075003000000240000
ZIPPER, ANN                      ZIP-714000CLERICAL       04500000000000000020
                                 ZZZ-999999                   4911900379966020
```

APPENDIX

JOB SHEET 1: Sample Application, Sequential Update Program For Bibliography

The purpose of the sample program is to update an inventory file which has a sequential organization and exists either on magnetic-disk storage or on magnetic tape. The program reads input transactions from cards and processes them against the existing master file to create a new updated master file. The hierarchy chart for the program is shown in Figure JS 1-11, and a complete listing of the program is given in Figure JS 1-12.

The program should be studied carefully. No flowchart has been provided; the preparation of the flowchart has been left to the student as a means of studying the program. Another useful approach to understanding the logic is to work through it with pencil and paper using some sample data. Figure JS 1-13 is a listing of the old master file as it exists on the disk (or tape). Figure JS 1-14 is a listing of the card images of the transactions to be processed against that file. These listings may be used in tracing the logic of the program. After the sample program is run with the master file and transactions shown in JS 1-13 and JS 1-14, the new, updated master file looks like Figure JS 1-15.

The transaction report accompanying this update program lists every transaction that is input, with a comment to the right of any transaction containing an error indicating the last type of error detected. Any transaction containing errors does not update the master file. The report output of the run with the given data is shown in JS 1-16. Most errors have been deleted from the transaction deck in order to center attention on master file updating rather than on transaction editing, thus editing routines have been omitted.

The processing logic of the program generally follows that described in this Unit. Transactions contain a three-character transaction code, which can have the value 'ADD', 'CHG', or 'DEL'. It is very important to note that there may be more than one transaction per master record. When there are several transactions, they must be sequenced so that the operations occur in the intended order. The alphabetic codes are useful for this illustration, but they do not sort naturally in the correct transaction sequence. It is possible to program around this problem, but many installations prefer numeric codes, which can be assigned in the corrrect sorting order.

Note that the paragraph entitled MAST-REPT-RTN which is performed as part of "the master key less than transaction key" activity, is a program stub. Program stubs are often used by programmers in a top-down testing strategy to debug higher level routines without having to simultaneously test the details of lower-level routines. In this routine each new master record is simply displayed without the extensive formatting and controls which would normally be produced.

There are several switches used in the program to indicate processing conditions. READ-SWITCH is a level 77 data item which takes on the following values to indicate which file is to be read in the next processing cycle:

'B' Read both the master file and the transaction file
'M' Read the master file only
'T' Read the transaction file only

Notice how READ-SWITCH is used in READ-CONTROL-ROUTINE to indicate which file or files to read:

```
READ-CONTROL-ROUTINE.
  IF   READ-SWITCH EQUALS 'B'
         PERFORM READ-MASTER
         PERFORM READ-TRANSACTION
  ELSE
         IF  READ-SWITCH EQUALS 'M'
             PERFORM READ-MASTER
         ELSE
             PERFORM READ-TRANSACTION.
```

Notice also the places in the program where the value of READ-SWITCH is changed. It is initialized to 'B' in the working-storage section so that in the first pass through the program, both files are read. Its value is subsequently set to one of the three values at different points in the program depending on the type of processing that occurred (see Figure JS 1-12).

Another programmed switch used in the example is ERROR-FLAG. Each time a new transaction record is read, ERROR-FLAG is set to 'N'. If an error occurs in updating, ERROR-FLAG is set to 'Y'.

The value of ERROR-FLAG is used in two ways: PRINT-TRANSACTION prints it on a report. If an error has occurred, certain additional information is printed to show where and why it occurred. The program must know whether to print the error comments, and the value in ERROR-FLAG, at the time of printing, provides that information on the report.

Another programmed switch defined in the DATA DIVISION is called MAST-STOR. This switch is used, in conjunction with a temporary data area the same size as a master-file record called TEMP-MAST-STOR, for a special case of processing the master file. When an 'ADD' transaction is processed, it must be treated like a master-file record so that subsequent transactions are checked against it and the correct sequence of master-file records, including added records, is preserved. To accomplish this, the program moves the current master-file record to the temporary area TEMP-MAST-STOR and sets MAST-STOR to 'Y'. The new record is built from the 'ADD' transaction in the old master-file-record area. Thus subsequent transactions can be processed without any special coding. The advantage of this technique is that only the ADD-ROUTINE and the READ-MASTER routines "need to know" that the new master record is being created. Thus other routines can treat the newly created record just as if it were any other old master record. This technique illustrates the programming principle of information hiding. The number of routines which "need to know" are minimized, thus minimizing perturbations to those other routines. Finally, when the next master record is to be read, the module READ-MASTER checks the MAST-STOR-SWITCH, and if it equals 'Y', the record in TEMP-MAST-STOR is returned to the WS-OLD-MAST area and the switch set to 'N'. If MAST-STOR equals 'N', there is no record in the temporary area and the next master file record is read. For example:

```
READ-MASTER.
   IF  MAST-STOR EQUALS 'Y'
          MOVE 'N' TO MAST-STOR
          MOVE TEMP-MAST-STOR
          TO WS-OLD-MAST
   ELSE
          READ BIBLIOG-OLD-MASTER-FILE
          AT END DISPLAY 'ERR AT END ON MASTER'.
```

The handling of end-of-file conditions in this sample program follows the logic explained in this Unit. It is necessary to ensure that if either file runs out of records, the other file will be processed to completion. Normally, the transaction file will end before the master file, but it is possible for the master file to end first, as when several records are added to the end of the file. In this program, the end-of-file procedure for both files is an error if it occurs; because this means that by error, the program has tried to read past the last card in each file. The last card in each file is a special card called a trailer record. The purpose of the trailer record is to signal the end of the file and to enable orderly completion of the program. The trailer record has an artificial value in its key field such that it is the highest value (sorts last) of any key in the file. Thus, if the trailer record is read, this means that no other records could remain in this file. If the trailer record for one file has been read, the high values in its key insure that all the remaining records in the other file have key values which are smaller. This means that if the transaction file ends first, all remaining master file records are written, as is to the new master file, through WRITE-NEW-MASTER, since the condition that the master-file key is smaller than the transaction key is met. Similarly, if the master file ends first, all remaining transactions are added (if they are 'ADD's), since their keys are smaller than the master record trailer key.

Regardless of which file ends first, at some point the remaining file also reaches an end-of-file condition. Then the record key of that file is obtained from its trailer record. This trailer has the same key value as the trailer record on the other file. Processing logic now leads to the performance of EQUAL-ROUTINE, since now both keys are equal. After one file is at the trailer record, the EQUAL-ROUTINE is not processed unless both are, so it is sufficient to check one file in EQUAL-ROUTINE for the trailer key value of "ZZ 999.999". If equal, both files have been processed completely. The switch MORE-RECORDS is set to 'N' to indicate to MAIN-CONTROL-ROUTINE that the update process has been completed. Thus, the program is completely finished with input files. Next, the totaling is completed, printed as a footing to the transaction report, and the trailer record for the new master file is created by performing the WRITE-NEW-MASTER routine. Ordinarily the totals to the files would have been updated and written out with the new trailer record but a simplified approach is used.

Figure JS1-11: Hierarchy Chart for Sample Bibliography Update Program

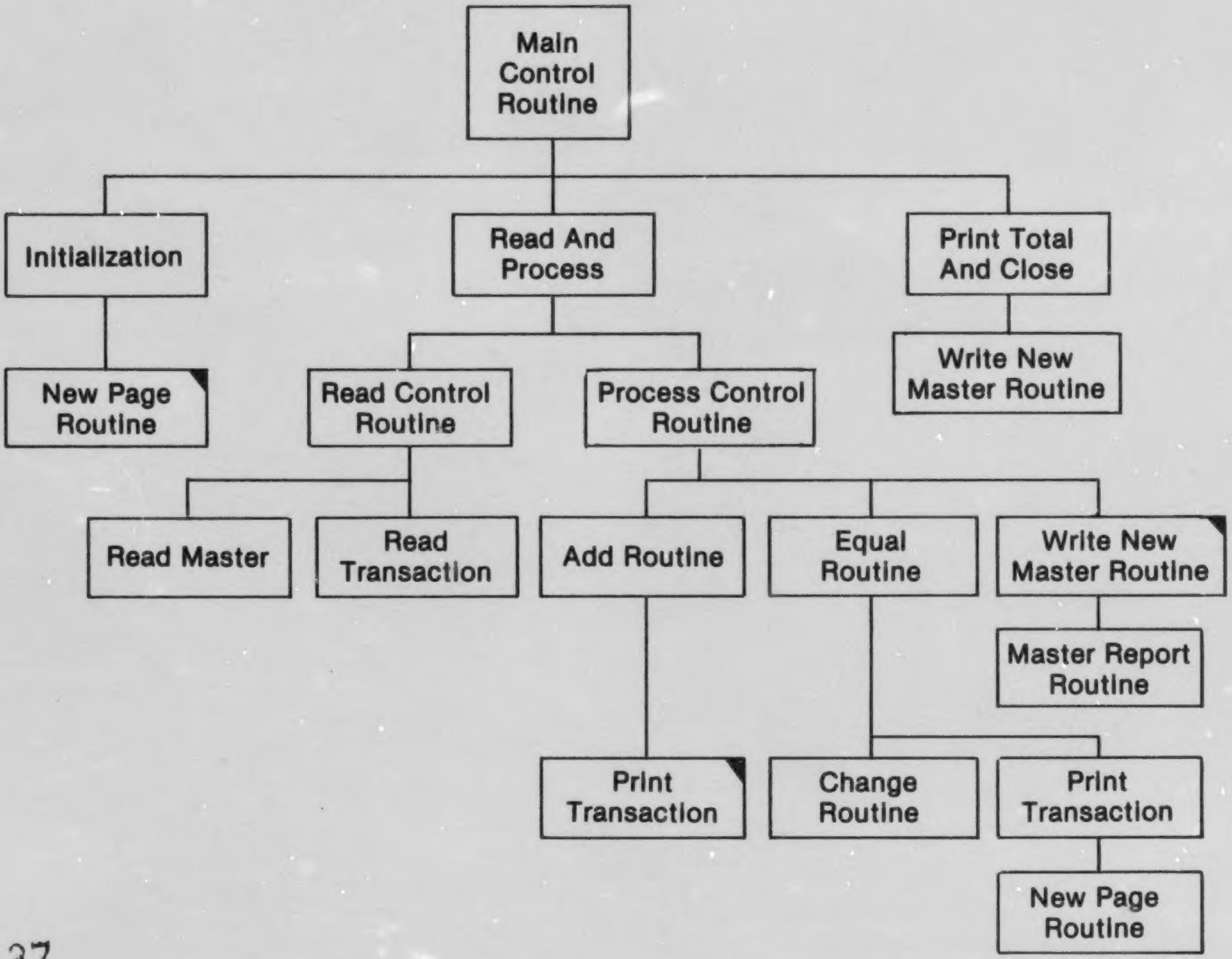

Main Control Routine
Initialization
Read And Process
Print Total And Close
New Page Routine
Read Control Routine
Process Control Routine
Write New Master Routine
Read Master
Read Transaction
Add Routine
Equal Routine
Write New Master Routine
Master Report Routine
Print Transaction
Change Routine
Print Transaction
New Page Routine

Figure JS1-12: Listing of Sample File Update Program

© The Curators of the
University of Missouri
1984, All Rights Reserved

```
 IDENTIFICATION DIVISION.
*
 PROGRAM-ID. BIBLIOG.
 AUTHOR. REVISED BY CHUCK LITECKY.
*
 ENVIRONMENT DIVISION.
*
 CONFIGURATION SECTION.
 SOURCE-COMPUTER. IBM-370.
 OBJECT-COMPUTER. IBM-370.
*
 SPECIAL-NAMES.
     C01 IS TOP-OF-PAGE.
*
 INPUT-OUTPUT SECTION.
 FILE-CONTROL.
     SELECT BIBLIOG-TRANSACTION-FILE ASSIGN TO UR-S-SYSIN.
     SELECT BIBLIOG-OLD-MASTER-FILE ASSIGN TO UR-DA-S-UMCSCR1.
     SELECT BIBLIOG-NEW-MASTER-FILE ASSIGN TO UR-DA-S-UMCSCR2.
     SELECT PRINT-FILE ASSIGN TO UR-S-SYSPRINT.
*
 DATA DIVISION.
*
 FILE SECTION.
*
 FD BIBLIOG-OLD-MASTER-FILE
     LABEL RECORD IS STANDARD.
 01  BIBLIOG-OLD-MASTER-RECORD PIC X(80).
*
 FD BIBLIOG-NEW-MASTER-FILE
     LABEL RECORD IS STANDARD.
 01  BIBLIOG-NEW-MASTER-RECORD       PIC X(80).
*
 FD  BIBLIOG-TRANSACTION-FILE
     LABEL RECORD IS OMITTED.
 01  BIBLIOG-TRANSACTION-RECORD PIC X(80).
*
 FD PRINT-FILE
     LABEL RECORD IS OMITTED.
 01  PRINT-LINE              PIC X(132).
*
 WORKING-STORAGE SECTION.
*
*        THERE ARE FOUR PROGRAMMED SWITHCHES USED IN THIS PROGRAM.
*        THE POSSIBLE VALUES FOR EACH AND THE MEANING OF EACH ARE:
*
*****    READ-SWITHCH INDICATES WHICH FILE OR FILES ARE TO BE READ
*        IN THE NEXT CYCLE OF THE PROGRAM. BASED ON THE RESULTS OF
*        SOME ACTIVITIY IN THE CURRENT CYCLE.  IT HAS 3 POSSIBLE
*        VALUES:
*            READ-SWITCH = 'T'
*                THIS SIGNIFIES THAT ON THE NEXT CYCLE, THE TRANS-
*                ACTION FILE IS TO BE READ.
*            READ-SWITCH = 'M'
*                ON THE NEXT CYCLE THE NEXT MASTER FILE RECORD IS
*                TO BE READ.
**           READ-SWITCH = 'B'
*                ON THE NEXT CYCLE THE NEXT RECORD EACH FILE IS
*                TO BE READ (THIS IS THE VALUE OF THE SWITCH THE
*                FIRST TIME THROUGH. AND ALSO AFTER DELETIONS).
*
*****    ERROR-FLAG SIGNALS WHETHER OR NOT A TRANSACTION RECORD
*        CONTAINS AN ERROR AFTER INPUT VALIDATION HAS BEEN
*        COMPLETED. IT HAS 2 POSSIBLE VALUES:
*            ERROR-FLAG = 'N'
*                THE TRANSACTION HAS BEEN VALIDATED AND CONTAINS
*                NO ERRORS. IT CAN THEREFORE BE PROCESSED AGAINST
*                THE MASTER FILE.
*            ERROR-FLAG = 'Y'
*                THE TRANSACTION CONTAINS ERRORS, AND THEREFORE
*                CANNOT BE PROCESSED AGAINST THE MASTER FILE.
*                PROCESSING WILL PASS TO READING THE NEXT TRANS-
*                ACTION.
*****    MAST-STOR IS A SPECIAL SWITCH TO INDICATE THAT
*        A MASTER RECORD HAS BEEN SAVED WHILE AN 'ADD' TRANSACTION
*        IS BEING PROCESSED. THE SWITCH SIGNALS THE NEXT ATTEMPT
*        TO READ A MASTER FILE RECORD THAT THE SAVED RECORD SHOULD
*        BE RESTORED INSTEAD. THIS METHOD IS EXPLAINED FURTHER IN
*                THE TEXT. THE SWITCH HAS TWO POSSIBLE VALUES:
*            MAST-STOR = 'N'
*                A REGULAR MASTER FILE RECORD SHOULD BE READ
*            MAST-STOR = 'Y'
*                THIS INDICATES THAT THERE IS A RECORD BEEING SAVED
*                AT MASTER FILE READ TIME. THIS RECORD SHOULD BE
*                RESTORED RATHER THAN A NEW MASTER RECORD READ.
*
*****    MORE-RECORDS IS AN END-OF-FILE SWITCH USED SIMILARLY TO
*        PREVIOUS PROGRAMS.  IN THIS CASE, IT IS SET TO INDICATE AN
*        END OF FILE CONDITION ONLY WHEN THE END OF BOTH FILES HAS
*        BEEN REACHED. ITS POSSIBLE VALUES:
*            MORE-RECORDS = 'Y'
*                AT LEAST ONE FILE CONTAINS RECORDS TO BE READ
*            MORE-RECORDS = 'N'
*                THE CLOSING PORION OF THE PROGRAM SHOULD BE
*                EXECUTED.
*
*        PLEASE NOTE BELOW THE FOLLOWING VALUES:
*
 77  LINE-COUNT              PIC 99.
 77  MORE-RECORDS            PIC X       VALUE 'Y'.
 77  READ-SWITCH             PIC X       VALUE 'B'.
 77  ERROR-FLAG              PIC X       VALUE 'N'.
 77  MAST-STOR               PIC X       VALUE 'N'.
*
 01  WS-OLD-MAST.
         05  MAST-KEY        PIC X(10).
         05  MAST-TITLE      PIC X(35).
```

Figure JS1-12: Listing of Sample File Update Program (cont'd)

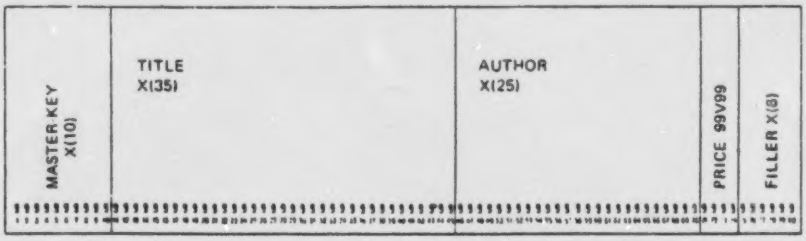
```
        05  MAST-AUTHOR      PIC X(25).
        05  MAST-PRICE       PIC 99V99.
        05  FILLER           PIC X(6).
01  WS-TRAN.
        05  TRANS-CODE       PIC X(3).
        05  TRANS-KEY        PIC X(10).
        05  TRANS-TITLE      PIC X(35).
        05  TRANS-AUTHOR     PIC X(25).
        05  TRANS-PRICE      PIC 99V99.
        05  FILLER           PIC X(3).
01  REPORT-TITLE.
    05 FILLER                PIC X(47)   VALUE SPACES.
    05  FILLER               PIC X(38)   VALUE
                    'BIBLIOGRAPHY UPDATE TRANSACTION REPORT'.
    05  FILLER               PIC X(47)   VALUE SPACES.
01  COLUMN-HEADING.
    05  FILLER               PIC XXX     VALUE SPACES.
    05  FILLER               PIC X(26)   VALUE
                    'CODE    KEY VALUE    TITLE'.
    05  FILLER               PIC X(32)   VALUE SPACES.
    05  FILLER               PIC X(6)    VALUE 'AUTHOR'.
    05  FILLER               PIC X(26)   VALUE SPACES.
    05  FILLER               PIC X(4)    VALUE 'COST'.
    05  FILLER               PIC X(15)   VALUE SPACES.
    05  FILLER               PIC X(14)   VALUE
                    'CAUSE OF ERROR'.
    05  FILLER               PIC X(6)    VALUE SPACES.
01  DETAIL-LINE.
    05  FILLER               PIC X(3)    VALUE SPACES.
    05  D-CODE               PIC X(3).
    05  FILLER               PIC X(5)    VALUE SPACES.
    05  D-KEY                PIC X(10).
    05  FILLER               PIC X(3)    VALUE SPACES.
    05  D-TITLE              PIC X(35).
    05  FILLER               PIC X(2)    VALUE SPACES.
    05  D-NAME               PIC X(25).
    05  FILLER               PIC X(5)    VALUE SPACES.
    05  D-PRICE              PIC $$9.99.
    05 FILLER                PIC X(5)    VALUE SPACES.
    05  ERROR-MESSAGE        PIC X(7).
    05  FILLER               PIC X(3)    VALUE SPACES.
    05  CAUSE-OF-ERROR-MESSAGE  PIC X(20).
01  TOTAL-LINE.
    05  FILLER               PIC X(20)   VALUE SPACES.
    05  FILLER               PIC X(29)   VALUE
                    'TOTAL NUMBER OF TRANSACTIONS:'.
    05  TL-TRANS-NUMBER      PIC ZZZZ9.
    05  FILLER               PIC X(10)   VALUE SPACES.
    05  FILLER               PIC X(13)   VALUE
                    'NUMBER RIGHT:'.
    05  TL-CORRECT-TRANS     PIC ZZZZ9.
    05  FILLER               PIC X(10)   VALUE SPACES.
    05  FILLER               PIC X(14)   VALUE
                    'NUMBER ERRORS:'.
    05  TL-WRONG-TRANS       PIC ZZZZ9.
```

```
    05  FILLER               PIC X(21)   VALUE SPACES.
01  TOTAL-COUNT.
    05  TC-TRANSACTION       PIC 9999    VALUE ZERO.
    05  TC-CORRECT           PIC 9999    VALUE ZERO.
    05  TC-WRONG             PIC 9999    VALUE ZERO.
01  TEMP-MAST-STOR           PIC X(80).
*
 PROCEDURE DIVISION.
*
 MAIN-CONTROL-ROUTINE.
     PERFORM INITIALIZATION.
     PERFORM READ-AND-PROCESS UNTIL MORE-RECORDS EQUAL 'N'
     PERFORM PRINT-TOTAL-AND-CLOSE.
     STOP RUN.
*
 INITIALIZATION.
     PERFORM MAST-STUB.
     OPEN INPUT BIBLIOG-OLD-MASTER-FILE.
     OPEN OUTPUT BIBLIOG-NEW-MASTER-FILE
         PRINT-FILE.
     PERFORM NEW-PAGE-ROUTINE.
*
 MAST-STUB.
     OPEN OUTPUT BIBLIOG-OLD-MASTER-FILE.
     OPEN INPUT BIBLIOG-TRANSACTION-FILE.
     PERFORM MAST-STUN UNTIL MORE-RECORDS EQUAL 'N'.
     MOVE 'Y' TO MORE-RECORDS.
     CLOSE BIBLIOG-OLD-MASTER-FILE.
 MAST-STUN.
     READ BIBLIOG-TRANSACTION-FILE AT END DISPLAY
      'ALL RECS READ IN WO HIGH VALUE TRAILER CARD'.
     MOVE  BIBLIOG-TRANSACTION-RECORD TO WS-OLD-MAST.
     IF   MAST-KEY EQUAL 'ZZ 999.999' MOVE 'N' TO MORE-RECORDS.
     DISPLAY WS-OLD-MAST.
     WRITE  BIBLIOG-OLD-MASTER-RECORD FROM WS-OLD-MAST.
 READ-AND-PROCESS.
     MOVE 'N' TO ERROR-FLAG.
     MOVE SPACES TO DETAIL-LINE.
     PERFORM READ-CONTROL-ROUTINE.
     PERFORM PROCESS-CONTROL-ROUTINE.
*
 READ-CONTROL-ROUTINE.
     IF READ-SWITCH EQUAL 'B'
         PERFORM READ-MASTER
         PERFORM READ-TRANSACTION
     ELSE
         IF READ-SWITCH EQUAL 'M'
             PERFORM READ-MASTER
         ELSE
             PERFORM READ-TRANSACTION.
*
 READ-MASTER.
     IF MAST-STOR EQUAL 'Y'
         MOVE 'N' TO MAST-STOR
         MOVE TEMP-MAST-STOR TO WS-OLD-MAST
```

Figure JS1-12: Listing of Sample File Update Program (cont'd)

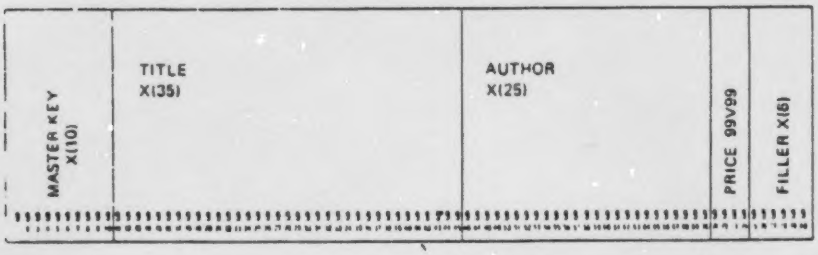
```
     ELSE
         READ BIBLIOG-OLD-MASTER-FILE INTO WS-OLD-MAST
         AT END DISPLAY 'ERR AT END OF MASTER'.
*
 READ-TRANSACTION.
     READ BIBLIOG-TRANSACTION-FILE INTO WS-TRAN
         AT END DISPLAY 'ERR AT END OF TRANS FILE'.
*
 PROCESS-CONTROL-ROUTINE.
     IF TRANS-KEY IS LESS THAN MAST-KEY
         PERFORM ADD-ROUTINE
     ELSE
         IF TRANS-KEY EQUAL MAST-KEY
             PERFORM EQUAL-ROUTINE
         ELSE
             PERFORM WRITE-NEW-MASTER.
*
 ADD-ROUTINE.
     IF TRANS-CODE NOT EQUAL TO 'ADD'
         MOVE 'Y' TO ERROR-FLAG
         MOVE 'NO MASTR REC FOUND' TO CAUSE-OF-ERROR-MESSAGE
     ELSE
         MOVE BIBLIOG-OLD-MASTER-RECORD TO TEMP-MAST-STOR
         MOVE 'Y' TO MAST-STOR
         MOVE TRANS-KEY TO MAST-KEY
         MOVE TRANS-TITLE TO MAST-TITLE
         MOVE TRANS-AUTHOR TO MAST-AUTHOR
         MOVE TRANS-PRICE TO MAST-PRICE.
     MOVE 'T' TO READ-SWITCH.
     PERFORM PRINT-TRANSACTION.
*
 EQUAL-ROUTINE.
     IF MAST-KEY EQUAL  'ZZ 999.999'
         MOVE 'N' TO MORE-RECORDS
     ELSE
         IF TRANS-CODE EQUAL 'DEL'
             MOVE 'B' TO READ-SWITCH
         ELSE
             IF TRANS-CODE EQUAL 'CHG'
                 PERFORM CHANGE-ROUTINE
                 MOVE 'T' TO READ-SWITCH
             ELSE
                 MOVE 'MASTR RECORD EXISTS' TO
                     CAUSE-OF-ERROR-MESSAGE
                 MOVE 'Y' TO ERROR-FLAG
                 MOVE 'T' TO READ-SWITCH.
     IF MORE-RECORDS EQUAL 'Y'
         PERFORM PRINT-TRANSACTION.
*
 CHANGE-ROUTINE.
     IF TRANS-TITLE NOT EQUAL SPACES
         MOVE TRANS-TITLE TO MAST-TITLE.
     IF TRANS-AUTHOR NOT EQUAL SPACE
         MOVE TRANS-AUTHOR TO MAST-AUTHOR.
     IF TRANS-PRICE NOT EQUAL SPACES
         MOVE TRANS-PRICE TO MAST-PRICE.
*
 WRITE-NEW-MASTER.
     WRITE BIBLIOG-NEW-MASTER-RECORD FROM WS-OLD-MAST.
     PERFORM MAST-REPT-RTN.
     MOVE 'M' TO READ-SWITCH.
*
 PRINT-TRANSACTION.
     IF LINE-COUNT GREATER THAN 50
         PERFORM NEW-PAGE-ROUTINE.
     MOVE TRANS-CODE TO D-CODE.
     MOVE TRANS-KEY TO D-KEY.
     MOVE TRANS-TITLE TO D-TITLE.
     MOVE TRANS-AUTHOR TO D-NAME.
     MOVE TRANS-PRICE TO D-PRICE.
     IF ERROR-FLAG EQUAL 'Y'
         MOVE '*ERROR*' TO ERROR-MESSAGE.
     WRITE PRINT-LINE FROM DETAIL-LINE
         AFTER ADVANCING 2 LINES.
     ADD 2 TO LINE-COUNT.
     ADD 1 TO TC-TRANSACTION.
     IF ERROR-FLAG EQUAL 'N'
         ADD 1 TO TC-CORRECT
     ELSE
         ADD 1 TO TC-WRONG.
*
 NEW-PAGE-ROUTINE.
     MOVE SPACES TO PRINT-LINE.
     WRITE PRINT-LINE AFTER ADVANCING TOP-OF-PAGE.
     WRITE PRINT-LINE FROM REPORT-TITLE AFTER ADVANCING 2 LINES.
     WRITE PRINT-LINE FROM COLUMN-HEADING AFTER ADVANCING 2 LINES.
     MOVE SPACES TO PRINT-LINE.
     MOVE 6 TO LINE-COUNT.
*
 PRINT-TOTAL-AND-CLOSE.
     MOVE TC-TRANSACTION TO TL-TRANS-NUMBER.
     MOVE TC-CORRECT TO TL-CORRECT-TRANS.
     MOVE TC-WRONG TO TL-WRONG-TRANS.
     WRITE PRINT-LINE FROM TOTAL-LINE
         AFTER ADVANCING 2 LINES.
     PERFORM WRITE-NEW-MASTER.
     CLOSE BIBLIOG-OLD-MASTER-FILE,
         BIBLIOG-TRANSACTION-FILE,
         BIBLIOG-NEW-MASTER-FILE,
         PRINT-FILE.
*
 MAST-REPT-RTN.
     DISPLAY 'MASTER IS ' WS-OLD-MAST.
*    THIS IS AN EXAMPLE OF A PROGRAM STUB
*     ORDINARILY THIS RTN WOULD PREPARE A REPORT
*        OF THE CONTENTS OF THE NEW MASTER FILE
*        AND ANY ASSOCIATED CONTROL INFO
*    BUT TO MAKE IT SIMPLER AND AID DEBUGGING, IT JUST DISPLAYS.
*    LATER THE PROGRAMMER WOULD ADD THE ACTUAL MASTER REPORT RTN.
```

Figure JS1-13: File Layout and Listing of Old Master File for Sample Program



```
BA 111.69DATA TRANSMISSION                   BACON, M. D.             1050
BA 157.234ASSEMBLERS AND LOADER              BARRON, D. W.            0925
BE 157.676GAMES PLAYING WITH COMPUTERS       BELL                     0800
BO 694.923MACHINE TRANSLATION                BOOTH, A. D.             2995
BR 921.596COMPUTER PROGRAMMING AND FORMAL SYSBRAFFORT, P. D.          0875
BR 922.749PROGRAM DEBUGGING                  BROWN, A. R.             0850
BR 923.464ORGANIZATION                       BROWN                    1350
CA 167.294AN INTRODUCTION TO MACROS          CAMPBEL-KELLY, M.        2695
CH 596.467COMPUTERS IN MATHEMATICAL RESEARCH CHURCHHOUSE, R. F.       2900
CO 666.666QUICK COBOL                        CODDINGTON               3550
CO 679.496INTRODUCTION TO OPERATING SYSTEMS  COLIN, A.                1175
DO 642.197DATABASE DESCRIPTION               DOUQUE, B. C. M.         1050
FO 762.987AUTOMATIC SYNTACTIC ANALYSIS       FOSTER, J. M.            0750
FO 784.123LIST PROCESSING                    FOSTER, J. M.            0675
FR 888.725ECONOMICS OF INFORMATICS           FRIELINK, A. B.          4175
HE 496.295DEVELOPMENTAL SYSTEMS AND LANGUAGESHERMAN, G. T.            3125
HO 743.656COMPILING TECHNIQUES               HOPGOOD, F. R. A.        6475
IG 457.967OPERATIONAL RES. IN DECISION MAKINGIGNIZIO                  1750
IL 523.796BASIC MACHINE PRINCIPLES           ILIFFE, J. K.            1875
JA 146.129DATA BASE MANAGEMENT SYSTEMS       JARDINE, D. A.           0850
KE 347.469HANDBOOK OF MANAGEMENT             KEMPNER                  2150
KL 597.267DATABASE MANAGEMENT                KIMBIE, J. W.            2995
LE 286.343COMPUTERS IN EDUCATION             LECARME, O. E.           1575
MA 176.545MAN AND COMPUTER                   MARDIS, M.               7495
MC 298.643CASES IN ORGANIZATIONAL BEHAVIOR   MCLENNEN                 2750
MU 895.743HUMAN CHOICE AND COMPUTERS         MUMFORD, E.              3550
NO 146.147GRAPHIC LANGUAGES                  NAKE, F.                 2295
NI 349.763AUTOMATA, LANGUAGES AND PROGRAMMINGNIVAT, M.                3750
PA 136.725QUEUING THEORY IN OPERATIONS RES.  PAGE                     1100
PE 293.476IMPLEMENTATION                     PECK, J. E. L.           2495
RO 643.242AN INTRO TO COMPILER WRITING       ROHL, J. S.              2095
RO 712.943INFORMATION PROCESSING             ROSENFELD, J. L.         7690
SC 22.451CONCEPTUAL INFORMATION PROCESSING   SCHANK, R. C.            2750
SC 297.675IMPLEMENTING OPER RES-MANAGE SC    SCHULTZ, R. L.           2475
ST 848.294ABSTRACT AUTOMATA                  STARKE, P. H.            2295
TR 946.543FINITE AUTOMATA                    TRAKHTENBROT, B. A.      2495
TU 998.476PROGRAMMING TEACHING TECHNIQUES    TURSKI, W. M.            1595
UN 547.184COMMAND LANGUAGES                  UMGER, C.                2995
VL 547.298COMPUTER-AIDED DESIGN              VLIETSTRA, J.            2750
WI 496.194TIME-SHARING COMPUTER SYSTEMS      WILKES, M.V.             1575
ZZ 999.999TRAILER RECORD
```

Figure JS1-14: Transaction Card Layout and List of Transactions Used in Sample Run for Sample Bibliography Update Program



```
ADD AB 146.923COMPUTER HARDWARE AND SOFTWARE        ABRAMS, M. D.              1295
DELBA 157.234
DELFO 762.987
CHGIL 523.796BASIC MACHINE PRINCIPLES
ADDJO 694.246COMPUTER SELECTION                     JOSLIN, E. O.              0795
ADDKN 557.462THE ART OF COMPUTER PROGRAMMING        KNUTH, D. E.               1995
DELLE 286.343
CHGMA 176.545                                                                  8000
CHGMA 176.545MAN AND HIS COMPUTER
CHGMU 895.743                                                                  4000
DELNA 146.147
DELPA 136.725
ADDSQ 778.986THE COMPUTER, AN EVERYDAY MACHINE      SQUIRE, E                  0587
DELTR 946.543
```

Figure JS1-15: Output for Sample Run of Biblio-Update Program Combined with File Contents

```
                                              BIBLIOGRAPHY UPDATE TRANSACTION REPORT
  CODE    KEY VALUE    TITLE                                AUTHOR                         COST               CAUSE OF ERROR

  ADD      AB 146.92   3COMPUTER HARDWARE AND SOFTWARE      ABRAMS, M. D.                $12.95
MASTER IS AB 146.923COMPUTER HARDWARE AND SOFTWARE     ABRAMS, M. D.            1295
MASTER IS BA 111.690DATA TRANSMISSION                  BACON, M. D.             1050

  DEL     BA 157.234                                                                      $0.00
MASTER IS BE 157.676GAMES PLAYING WITH COMPUTERS       BELL                     0800
MASTER IS BO 694.923MACHINE TRANSLATION                BOOTH, A. D.             2995
MASTER IS BR 921.596COMPUTER PROGRAMMING AND FORMAL SYSBRAFFORT, P. D.          0875
MASTER IS BR 922.749PROGRAM DEBUGGING                  BROWN, A. R.             0850
MASTER IS BR 923.464ORGANIZATION                       BROWN                    1350
MASTER IS CA 167.294AN INTRODUCTION TO MACROS          CAMPBEL-KELLY, M.        2695
MASTER IS CH 596.467COMPUTERS IN MATHEMATICAL RESEARCH CHURCHHOUSE, R. F.       2900
MASTER IS CO 666.666QUICK COBOL                        CODDINGTON               3550
MASTER IS CO 679.496INTRODUCTION TO OPERATING SYSTEMS  COLIN, A.                1175
MASTER IS DO 642.197DATABASE DESCRIPTION               DOUQUE, B. C. M.         1050

  DEL     FO 762.987                                                                      $0.00
MASTER IS FO 784.123LIST PROCESSING                    FOSTER, J. M.            0675
MASTER IS FR 888.725ECONOMICS OF INFORMATICS           FRIELINK, A. B.          4175
MASTER IS HE 496.295DEVELOPMENTAL SYSTEMS AND LANGUAGESHERMAN, G. T.            3125
MASTER IS HO 743.656COMPILING TECHNIQUES               HOPGOOD, F. R. A.        6475
MASTER IS IG 457.967OPERATIONAL RES. IN DECIAION MAKINGIGNIZIO                  1750

  CHG     IL 523.796   BASIC MACHINE PRINCIPLES                                           $0.00
MASTER IS IL 523.796BASIC MACHINE PRINCIPLES           ILIFFE, J. K.            1875
MASTER IS JA 146.129DATA BASE MANAGEMENT SYSTEMS       JARDINE, D. A.           0850

  ADD     JO 694.246   COMPUTER SELECTION                   JOSLIN, E. O.                 $7.95
MASTER IS JO 694.246COMPUTER SELECTION                 JOSLIN, E. D.            0795
MASTER IS KE 347.469HANDBOOK OF MANAGEMENT             KEMPNER                  2150
MASTER IS KL 597.267DATABASE MANAGEMENT                KIMBIE, J. W.            2995

  ADD     KN 557.462   THE ART OF COMPUTER PROGRAMMING      KNUTH, D. E.                 $19.95
MASTER IS KN 557.462THE ART OF COMPUTER PROGRAMMING    KNUTH, D. E.             1995

  DEL     LE 286.343                                                                      $0.00

  CHG     MA 176.545                                                                     $80.00

  CHG     MA 176.545   MAN AND HIS COMPUTER                                               $0.00
MASTER IS MA 176.545MAN AND HIS COMPUTER               MARDIS, M.               8000
MASTER IS MC 298.643CASES IN ORGANIZATIONAL BEHAVIOR   MCLENNEN                 2750

  CHG     MU 895.743                                                                     $40.00
MASTER IS MU 895.743HUMAN CHOICE AND COMPUTERS         MUMFORD, E.              4000

  DEL     NA 146.147                                                                      $0.00     *ERROR*   NO MASTR REC FOUND
MASTER IS NO 146.147GRAPHIC LANGUAGES                  NAKE, F.                 2295
MASTER IS NI 349.763AUTOMATA, LANGUAGES AND PROGRAMMINGNIVAT, M.                3750

  DEL     PA 136.725                                                                      $0.00
MASTER IS PE 293.476IMPLEMENTATION                     PECK, J. E. L.           2495
MASTER IS RO 643.242AN INTRO TO COMPILER WRITING       ROHL, J. S.              2095
MASTER IS RO 712.943INFORMATION PROCESSING             ROSENFELD, J. L.         7690
```

Figure JS1-15: Output for Sample Run of Biblio-Update Program
Combined with File Contents (cont'd)

```
MASTER IS SC 22.451CONCEPTUAL INFORMATION PROCESSING   SCHANK. R. C.              2750
MASTER IS SC 297.675IMP_EMENTING OPER RES-MANAGE SC    SCHULTZ. R. L.             2475

  ADD     SQ 778.986   THE COMPUTER. AN EVERYDAY MACHINE    SQUIRE, E                      $5.87
MASTER IS SQ 778.986THE COMPUTER. AN EVERYDAY MACHINE  SQUIRE. E                  0587
MASTER IS ST 848.294ABSTRACT AUTOMATA                  STARKE. P. H.              2295

  DEL     TR 946.543                                                                       $0.00
MASTER IS TU 998.476PROGRAMMING TEACHING TECHNIQUES    TURSKI. W. M.              1595

  CHG     UN 547.184                                        UNGER. C.                      $0.00
MASTER IS UN 547.184COMMAND LANGUAGES                  UNGER, C.                  2995
MASTER IS VL 547.298COMPUTER-AIDED DESIGN              VLIETSTRA. J.              2750

  DEL     WI 496.194                                                                       $0.00

  ADD     WI 499.246   THE REAL COMPUTER                    WIHTINGTON. F. G.              $9.95
MASTER IS WI 499.246THE REAL COMPUTER                  WIHTINGTON. F. G.          0995
                   TOTAL NUMBER OF TRANSACTIONS:    17         NUMBER RIGHT:    16         NUMBER ERRORS:
MASTER IS ZZ 999.999TRAILER RECORD
```

Figure JS1-16: Bibliography Update Transaction Report

```
                                      BIBLIOGRAPHY UPDATE TRANSACTION REPORT
CODE    KEY VALUE    TITLE                               AUTHOR                 COST               CAUSE OF ERROR
ADD      AB 146.92   3COMPUTER HARDWARE AND SOFTWARE     ABRAMS, M. D.        $12.95
DEL     BA 157.234                                                             $0.00
DEL     FO 762.987                                                             $0.00
CHG     IL 523.796   BASIC MACHINE PRINCIPLES                                  $0.00
ADD     JO 694.246   COMPUTER SELECTION                  JOSLIN, E. O.         $7.95
ADD     KN 557.462   THE ART OF COMPUTER PROGRAMMING     KNUTH, D. E.         $19.95
DEL     LE 286.343                                                             $0.00
CHG     MA 176.545                                                            $30.00
CHG     MA 176.545   MAN AND HIS COMPUTER                                      $0.00
CHG     MU 895.743                                                            $40.00
DEL     NA 146.147                                                             $0.00     *ERROR*   NO MASTR REC FOUND
DEL     PA 136.725                                                             $0.00
ADD     SQ 778.986   THE COMPUTER, AN EVERYDAY MACHINE   SQUIRE, E             $5.87
DEL     TR 946.543                                                             $0.00
CHG     UN 547.184                                       UNGER, C.             $0.00
DEL     WI 496.194                                                             $0.00
ADD     WI 499.246   THE REAL COMPUTER                   WIHTINGTON, F. G.     $9.95
              TOTAL NUMBER OF TRANSACTIONS:   17          NUMBER RIGHT:   16          NUMBER ERRORS:    1
```

Answer to Job Sheet 1: Complete Program Solution to JS1, H1, H2, and H3 Combined

```
 IDENTIFICATION DIVISION.
*
 PROGRAM-ID. PAYROL.
*
 ENVIRONMENT DIVISION.
*
 CONFIGURATION SECTION.
 SOURCE-COMPUTER. IBM-370.
 OBJECT-COMPUTER. IBM-370.
*
 SPECIAL-NAMES.
     C01 IS TOP-OF-PAGE.
*
 INPUT-OUTPUT SECTION.
 FILE-CONTROL.
     SELECT PRINT-FILE ASSIGN TO UR-S-SYSPRINT.
     SELECT E-TRANS-FILE ASSIGN TO UR-S-SYSIN.
     SELECT O-MAST-FILE ASSIGN TO UR-DA-S-SCRFILE1.
     SELECT N-MAST-FILE ASSIGN TO UR-DA-S-SCRFILE2.
     SELECT M-SPOOL-FILE ASSIGN TO UR-DA-S-SCRFILE3.
*
 DATA DIVISION.
*
  FILE SECTION.
*
 FD  O-MAST-FILE
     LABEL RECORD IS OMITTED.
 01  O-MAST-REC                    PIC X(80).
*
 FD  N-MAST-FILE
     LABEL RECORD IS OMITTED.
 01  N-MAST-REC                    PIC X(80).
*
 FD  E-TRANS-FILE
     LABEL RECORD IS OMITTED.
 01  E-TRAN-REC                    PIC X(80).
*
 FD  PRINT-FILE
     LABEL RECORD IS OMITTED.
 01  PRINT-LINE                    PICTURE X(132).
*
 FD  M-SPOOL-FILE
     LABEL RECORD IS OMITTED.
 01  M-SPOOL-REC                  PIC X(132).
*
 WORKING-STORAGE SECTION.
*
 77  LINE-COUNT                    PICTURE 99       VALUE 60.
 77  READ-SWITCH                   PICTURE X        VALUE 'B'.
*B=BOTH FILES,M=MASTER ONLY,T=TRANSACTION ONLY.
 77  ERROR-FLAG                    PICTURE X        VALUE 'N'.
*N=NO ERROR,Y=YES THERE IS AN ERROR.
 77  MASTER-STORE-SWITCH           PICTURE X        VALUE 'N'.
*N=NO MASTER STORED,Y=YES THERE IS A MASTER STORED.
 77  MORE-RECORDS                  PICTURE X        VALUE 'Y'.
*Y=YES THERE ARE  MORE RECORDS,N=NO MORE RECORDS.
*
 01  WS-MAST-REC.
     05  O-EMP-NAME                PIC X(25).
     05  O-EMP-ID                  PIC X(10).
     05  O-DEPT                    PIC X(15).
     05  O-PAY-RATE                PIC 99V99.
     05  O-GPAY-YTD                PIC 99999V99.
     05  O-NPAY-YTD                PIC 99999V99.
     05  FILLER                    PICTURE X(12).
*
 01  WS-TRAN-REC.
     05  E-EMP-NAME                PIC X(25).
     05  E-EMP-ID                  PIC X(10).
     05  E-DEPT                    PIC X(15).
     05  E-PAY-RATE                PIC 99V99.
     05  E-GPAY                    PIC 9999V99.
     05  E-NPAY                    PIC 9999V99.
     05  FILLER                    PICTURE X(11).
     05  T-CODE                    PIC XXX.
*
 01  REPORT-TITLE.
     05  FILLER                    PICTURE X(50)   VALUE SPACE
     05  FILLER                    PICTURE X(33)   VALUE
           'PAYROLL UPDATE TRANSACTION REPORT'.
     05  FILLER                    PICTURE X(49)   VALUE SPACE
```

```
*
 01  HEAD1.
     05  FILLER                 PICTURE X(4)    VALUE SPACES.
     05  FILLER                 PICTURE X(5)    VALUE 'TRANS'
     05  FILLER                 PICTURE X(72)   VALUE SPACES.
     05  FILLER                 PICTURE X(9)    VALUE
                           'RATE     '.
     05  FILLER                 PICTURE X(20)   VALUE
                           'GROSS PAY    NET PAY'.
     05  FILLER                 PICTURE X(22)   VALUE SPACES.
*
 01  HEAD2.
     05  FILLER                 PICTURE X(4)    VALUE SPACES.
     05  FILLER                 PICTURE X(4)    VALUE 'CODE'.
     05  FILLER                 PICTURE X(5)    VALUE SPACES.
     05  FILLER                 PICTURE X(15)   VALUE
                           'ID NUMBER      '.
     05  FILLER                 PICTURE X(30)   VALUE
                           'EMPLOYEE NAME                 '.
     05  FILLER                 PICTURE X(22)   VALUE
                           'DEPARTMENT            '.
     05  FILLER                 PICTURE X(30)   VALUE
                           'OF PAY      THIS PD      THIS PD'.
     05  FILLER                 PICTURE X(22)   VALUE
                           '  CAUSE OF ERROR      '.
*
 01  D-DETAIL-LINE.
     05  FILLER                 PICTURE X(5)    VALUE SPACES.
     05  D-T-CODE              PIC XXX.
     05  FILLER                 PICTURE X(5)    VALUE SPACES.
     05  D-EMP-ID              PIC X(10).
     05  FILLER                 PICTURE X(5)    VALUE SPACES.
     05  D-EMP-NAME            PIC X(25).
     05  FILLER                 PICTURE X(5)    VALUE SPACES.
     05  D-DEPT                PIC X(15).
     05  FILLER                 PICTURE X(7)    VALUE SPACES.
     05  D-PAY-RATE            PIC $$9.99.
     05  FILLER                 PICTURE X(3)    VALUE SPACES.
     05  D-GPAY                PIC $$$$$$.99.
     05  FILLER                 PICTURE X(3)    VALUE SPACES.
     05  D-NPAY                PIC  $$$$$$.99.
     05  FILLER                 PICTURE X(2)    VALUE SPACES.
     05  CAUSE-OF-ERROR         PICTURE X(20).
*
 01  TOTAL-LINE.
     05  FILLER                 PICTURE X(5)   VALUE SPACES.
     05  FILLER                 PICTURE X(19)   VALUE
                           'TOTAL TRANSACTIONS:'.
     05  TOT-TRANS           PIC ZZZZ9.
     05  FILLER                 PICTURE X(15)   VALUE
                           '        CORRECT:'.
     05  TOT-RIGHT           PIC ZZZZ9.
     05  FILLER                 PICTURE X(14)   VALUE
                           '        ERRORS:'.
     05  TOT-WRONG           PIC ZZZZ9.
     05  FILLER                 PICTURE X(21)   VALUE
                           '     TOTAL GROSS PAY:'.
     05  TOT-GPAY            PIC $$$$$$$9.99.
     05  FILLER                 PICTURE X(12)   VALUE
                           '     NET PAY:'.
     05  TOT-NPAY            PIC $$$$$$$9.99.
     05  FILLER                 PICTURE X(6)    VALUE SPACES.
*
 01  T-TOTALS.
     05  T-TOT-TRANS            PIC 9999        VALUE ZERO.
     05  T-TOT-RIGHT            PIC  9999       VALUE ZERO.
     05  T-TOT-WRONG            PIC 9999        VALUE ZERO.
     05  T-TOT-GPAY             PIC 9999999V99  VALUE ZERO.
     05 T-TOT-NPAY              PIC 9999999V99  VALUE ZERO.
*
 01  TEMP-MAST-STORE            PIC X(84).
*
*
 01  S-NMASTR-TRAILER.
     05  FILLER                    PIC X(25).
     05  S-EMP-ID                    PIC X(10) VALUE IS
                                   'ZZZ-999999'.
     05  FILLER                    PIC X(19).
     05  S-GP                      PIC 9(5)V99 VALUE IS ZERO.
     05  S-NP                      PIC 9(5)V99 VALUE IS ZERO.
     05  S-NR                      PIC 9(5)V99 VALUE IS ZERO.
     05  FILLER                    PIC X(5).
```

Answer to Job Sheet 1: Complete Program Solution to JS1, H1, H2, and H3 Combined

```
*
 01  C-TRAN-TOTALS.
     05  C-EMP-ID                   PIC X(10) VALUE IS
                                  'ZZZ-999999'.
     05  C-GP                     PIC 9(5)V99 VALUE IS ZERO.
     05  C-NP                     PIC 9(5)V99 VALUE IS ZERO.
     05  C-NR                     PIC 9(5)V99 VALUE IS ZERO.
*
 01  F-OMAST-TOTALS.
     05  F-EMP-ID                   PIC X(10) VALUE IS
                                  'ZZZ-999999'.
     05  F-GP                     PIC 9(5)V99 VALUE IS ZERO.
     05  F-NP                     PIC 9(5)V99 VALUE IS ZERO.
     05  F-NR                     PIC 9(5)V99 VALUE IS ZERO.
*
 01  M-HEAD-1.
     05  FILLER                   PIC X(56) VALUE SPACE.
     05  FILLER                   PIC X(21) VALUE
         'PAYROLL MASTER REPORT'.
     05  FILLER                   PIC X(55) VALUE SPACE.
*
 01  M-HEAD-2.
     05      FILLER               PIC X(5) VALUE SPACE.
     05      FILLER               PIC X(7) VALUE
             'ID     '.
     05      FILLER               PIC X(10) VALUE SPACE.
     05      FILLER               PIC X(13) VALUE
             'EMPLOYEE NAME'.
     05      FILLER               PIC X(10) VALUE SPACE.
     05      FILLER               PIC X(25) VALUE
             'DEPARTMENT'.
     05      FILLER               PIC X(4) VALUE SPACE.
     05      FILLER               PIC X(9) VALUE
             'PAY RATE'.
     05      FILLER               PIC X(12) VALUE SPACE.
     05      FILLER               PIC X(9) VALUE
             'GROSS PAY'.
     05      FILLER               PIC X(11) VALUE SPACE.
     05      FILLER               PIC X(7) VALUE
             'NET PAY'.
     05      FILLER               PIC X(10) VALUE SPACE.
 01      M-DETAIL.
     05      FILLER               PIC X(3).
     05      M-ID                 PIC X(10).
     05      FILLER               PIC X(9).
     05      M-NAME               PIC X(15).
     05      FILLER               PIC X(8).
     05      M-DEPT               PIC X(25).
     05      FILLER               PIC X(5).
     05      M-PAY-RATE           PIC 999.99.
     05      FILLER               PIC X(13).
     05      M-GPAY               PIC ZZZZZZ.99.
     05      FILLER               PIC X(10).
     05      M-NPAY               PIC ZZZZZZ.99.
     05      FILLER               PIC X(10).
*
 PROCEDURE DIVISION.
*
 MAIN-CONTROL-ROUTINE.
     PERFORM INITIALIZATION.
     PERFORM READ-AND-PROCESS UNTIL MORE-RECORDS = 'N'.
     PERFORM PRINT-TOTALS.
     PERFORM CLOSING.
     STOP RUN.
*
 INITIALIZATION.
     OPEN INPUT E-TRANS-FILE.
     OPEN OUTPUT O-MAST-FILE.
     PERFORM MASTER-CREATE-ROUTINE UNTIL MORE-RECORDS EQUAL 'N'.
     MOVE 'Y' TO MORE-RECORDS.
     CLOSE O-MAST-FILE.
     OPEN INPUT O-MAST-FILE.
     OPEN OUTPUT N-MAST-FILE, PRINT-FILE.
*
 MASTER-CREATE-ROUTINE.
     READ E-TRANS-FILE AT END DISPLAY
          'ALL RECORDS READ WITHOUT HIGH VALUES CARD'.
     MOVE E-TRAN-REC TO WS-MAST-REC.
     IF O-EMP-ID EQUAL 'ZZZ-999999' MOVE 'N' TO MORE-RECORDS.
     MOVE WS-MAST-REC TO O-MAST-REC.
     DISPLAY O-MAST-REC.
     WRITE O-MAST-REC.
```

```
*
 READ-AND-PROCESS.
     PERFORM READ-CONTROL.
     PERFORM PROCESS-CONTROL.
*
 READ-CONTROL.
     IF READ-SWITCH = 'B'
         PERFORM READ-MAST
         PERFORM READ-TRANS
     ELSE
         IF READ-SWITCH = 'M'
             PERFORM READ-MAST
         ELSE
             PERFORM READ-TRANS.
*
 READ-MAST.
     IF MASTER-STORE-SWITCH = 'Y'
         MOVE 'N' TO MASTER-STORE-SWITCH
         MOVE TEMP-MAST-STORE TO WS-MAST-REC
     ELSE
         READ O-MAST-FILE INTO WS-MAST-REC
             AT END DISPLAY 'ERROR, AT END OF MASTER'.
*
 READ-TRANS.
     READ E-TRANS-FILE INTO WS-TRAN-REC AT END DISPLAY
         'ERROR, AT END OF TRANSACTION FILE'.
*
 NEW-PAGE.
     IF LINE-COUNT > 50
         WRITE PRINT-LINE FROM REPORT-TITLE
             AFTER ADVANCING TOP-OF-PAGE
         WRITE PRINT-LINE FROM HEAD1
             AFTER ADVANCING 1 LINES
         WRITE PRINT-LINE FROM HEAD2
             AFTER ADVANCING 1 LINES
         MOVE 3 TO LINE-COUNT.
*
 PROCESS-CONTROL.
     IF E-EMP-ID < O-EMP-ID
         PERFORM ADD-RTN
     ELSE
         IF E-EMP-ID = O-EMP-ID
             PERFORM EQUAL-RTN
         ELSE
             PERFORM WRITE-MAST-RTN.
*
 ADD-RTN.
     IF T-CODE NOT EQUAL TO 'ADD'
         MOVE 'Y' TO ERROR-FLAG
         MOVE 'NO MASTER FOUND' TO CAUSE-OF-ERROR
     ELSE
        MOVE O-MAST-REC TO TEMP-MAST-STORE
        MOVE 'Y' TO MASTER-STORE-SWITCH
        MOVE E-EMP-NAME TO O-EMP-NAME
        MOVE E-EMP-ID TO O-EMP-ID
        MOVE E-DEPT TO O-DEPT
        MOVE E-PAY-RATE TO O-PAY-RATE
        MOVE E-GPAY TO O-GPAY-YTD
        MOVE E-NPAY TO O-NPAY-YTD
        MOVE 'T' TO READ-SWITCH.
     PERFORM PRINT-TRANS.
*
 PRINT-TRANS.
     IF LINE-COUNT > 50
        PERFORM NEW-PAGE.
        MOVE T-CODE TO D-T-CODE.
        MOVE E-EMP-NAME TO D-EMP-NAME.
        MOVE E-EMP-ID TO D-EMP-ID.
        MOVE E-DEPT TO D-DEPT.
        MOVE E-PAY-RATE TO D-PAY-RATE.
        MOVE E-GPAY TO O-GPAY.
        MOVE E-NPAY TO D-NPAY.
        WRITE PRINT-LINE FROM D-DETAIL-LINE
            AFTER ADVANCING 2 LINES.
        ADD 2 TO LINE-COUNT.
        ADD 1 TO T-TOT-TRANS.
        IF ERROR-FLAG = 'Y'
            ADD 1 TO T-TOT-WRONG
        ELSE
            ADD 1 TO T-TOT-RIGHT.
```

Answer to Job Sheet 1: Complete Program Solution to JS1, H1, H2, and H3 Combined

```
*
 EQUAL-RTN.
     IF O-EMP-ID = 'ZZZ-999999'
         PERFORM WRITE-MAST-RTN
     ELSE
         IF T-CODE = 'DEL'
             MOVE 'B' TO READ-SWITCH
         ELSE
             IF T-CODE = 'CHG'
                 PERFORM CHG-RTN
                 MOVE 'T' TO READ-SWITCH
             ELSE
                 IF T-CODE = 'PAY'
                     PERFORM PAY-RTN
                     MOVE 'T' TO READ-SWITCH
                 ELSE
                     MOVE 'MASTER RECORD EXISTS' TO CAUSE-OF-ERROR
                     MOVE 'Y' TO ERROR-FLAG
                     MOVE 'T' TO READ-SWITCH.
     IF MORE-RECORDS = 'Y'
         PERFORM PRINT-TRANS.
*
 CHG-RTN.
     IF E-EMP-ID NOT EQUAL SPACES
         MOVE E-EMP-ID TO O-EMP-ID.
     IF E-EMP-NAME NOT EQUAL SPACES
         MOVE E-EMP-NAME TO O-EMP-NAME.
     IF E-PAY-RATE NOT EQUAL SPACES
         MOVE E-PAY-RATE TO O-PAY-RATE.
```

```
*
 PAY-RTN.
     ADD E-GPAY TO O-GPAY-YTD, T-TOT-GPAY.
     ADD E-NPAY TO O-NPAY-YTD, T-TOT-NPAY.
*
 WRITE-MAST-RTN.
     IF O-EMP-ID = 'ZZZ-999999' MOVE 'N' TO MORE-RECORDS.
     WRITE N-MAST-REC FROM WS-MAST-REC.
     MOVE 'M' TO READ-SWITCH.
*
 WRITE-UPDATED-MAST.
     READ N-MAST-FILE INTO WS-MAST-REC.
     DISPLAY WS-MAST-REC.
     IF O-EMP-ID = 'ZZZ-999999' MOVE 'N' TO MORE-RECORDS.
*
 PRINT-TOTALS.
     MOVE T-TOT-TRANS TO TOT-TRANS.
     MOVE T-TOT-GPAY TO TOT-GPAY.
     MOVE T-TOT-NPAY TO TOT-NPAY.
     MOVE T-TOT-RIGHT TO TOT-RIGHT.
     MOVE T-TOT-WRONG TO TOT-WRONG.
     WRITE PRINT-LINE FROM TOTAL-LINE AFTER ADVANCING 4 LINES.
*
 CLOSING.
     CLOSE E-TRANS-FILE, PRINT-FILE, O-MAST-FILE, N-MAST-FILE.
     OPEN INPUT N-MAST-FILE.
     MOVE 'Y' TO MORE-RECORDS.
     PERFORM WRITE-UPDATED-MAST UNTIL MORE-RECORDS = 'N'.
     CLOSE N-MAST-FILE.
*
*
```

Data Files: Old Master and Updated Master

```
BURKE, SADIE             BUR-145927DATA PROCESSING039501580000126400
BURDEN, JACK             BUR-382196ACCOUNTING     073502940000220500
BUTLER, RHETT            BUT-824571MARKETING      087503500000262500
FAVIR, LARRY             FAV-835127MARKETING      070002800000210000
FINKBINE, GEORGE         FIN-571433ACCOUNTING     047501900000150000
HAMILTON, MELANIE        HAM-446735DATA PROCESSING038501732500138600
HAMPTON, WADE            HAM-642151DATA PROCESSING043201728000138240
IRWIN, MONTAGUE          IRW-924586ACCOUNTING     085603424000256800
MACINTOSH, ANGUS         MAC-662473MARKETING      053002226000178080
MASTERN, CASS            MAS-619318MARKETING      045001912500153000
MERRIWETHER, MAYBELLE    MER-527348DATA PROCESSING027501100000093500
MORRISON, JAMES          MOR-754884DATA PROCESSING057301942000156700
OHARA SCARLETT           OHA-431552MARKETING      045001800000144000
PORTNOY, ALEXANDER       POR-783914DATA PROCESSING100004000000300000
ROBILLARD, ELLEN         ROB-445331DATA PROCESSING036001440000115200
SLATTERY, THOMAS         SLA-108423ACCOUNTING     093503740000280500
STANTON, ANNE            STA-274432DATA PROCESSING090503740000280500
STARK, WILLIE            STA-367810MARKETING      060202528000202240
TARLETON, BRENT          TAR-311834DATA PROCESSING037501500000120000
WATLING, BELLE           WAT-623489MARKETING      063202528000189600
WYNDER, ABLE             WYN-246372ACCOUNTING     075003000000240000
                         ZZZ-999999                    4911900379966020
BURKE, SADIE             BUR-145927DATA PROCESSING039501738000139040
BURDEN, JOHN             BUR-382196ACCOUNTING     073502940000220500
BUTLER, RHETT            BUT-824571MARKETING      087503500000262500
CALVERT, CATHLEEN        CAL-336741MARKETING      06250     00     0
FINKBINE, GEORGE         FIN-571433ACCOUNTING     047502090000165200
HAMILTON, MELANIE        HAM-446735DATA PROCESSING038501732500138600
HAMPTON, WADE            HAM-642151DATA PROCESSING043201728000138240
MACINTOSH, ANGUS         MAC-662473MARKETING      053002226000178080
MASTERN, CASS            MAS-619318MARKETING      050002112500169000
MERRIWETHER, MAYBELLE    MER-527348DATA PROCESSING027501100000093500
MORRISON, JAMES          MOR-754884DATA PROCESSING057301942000156700
OHARA SCARLETT           OHA-431552MARKETING      045001800000144000
PORTNOY, ALEXANDER       POR-783914DATA PROCESSING100004000000300000
ROBILLARD, ELLEN         ROB-445331DATA PROCESSING049901440000115200
SLATTERY, THOMAS         SLA-108423ACCOUNTING     093504114000308550
STANTON, ANNE            STA-274432DATA PROCESSING090504140500310775
STARK, WILLIE            STA-367810MARKETING      060202528000202240
SUTTER, BRUCE            SUT-654321DATA PROCESSING15000     00     0
TARLETON, BRENT          TAR-311834DATA PROCESSING037501500000120000
WATLING, BELLE           WAT-623489MARKETING      063202528000189600
WILKES, ASHLEY           WIL-300177ACCOUNTING     08350     00     0
WYNDER, ABLE             WYN-246372ACCOUNTING     075003000000240000
ZIPPER, ANN              ZIP-714000CLERICAL       04500000000000000020
                         ZZZ-999999                    4911900379966020
```

Sample Correct Output for Program H2 of Job Sheet One

```
                                                PAYROLL UPDATE TRANSACTION REPORT
TRANS                                                                     RATE     GROSS PAY    NET PAY
CODE    ID NUMBER    EMPLOYEE NAME              DEPARTMENT              OF PAY     THIS PD      THIS PD    CAUSE OF ERROR

 PAY    BUR-145927                                                       $0.00     $158.00      $126.40

 CHG    BUR-382196   BURDEN, JOHN                                        $0.00        $.00         $.00

 ADD    CAL-336741   CALVERT, CATHLEEN          MARKETING                $6.25        $.00         $.00

 DEL    FAV-835127                                                       $0.00     $190.00      $152.00

 PAY    FIN-571433                                                       $0.00     $190.00      $152.00

 CHG    HAM-446735                              ACCOUNTING               $0.00        $.00         $.00

 DEL    IRW-924586   IRWIN, MONTAGUE            ACCOUNTING               $8.56     $342.40       $25.68

 CHG    MAS-619318                              PRODUCTION               $5.00        $.00         $.00

 PAY    MAS-619318                                                       $0.00     $200.00      $160.00

 CHG    ROB-445331                                                       $4.99     $318.00      $254.00

 PAY    SLA-108423                                                       $0.00     $374.00      $280.50

 PAY    STA-274432                                                       $9.05     $400.50      $302.75

 ADD    SUT-654321   SUTTER, BRUCE              DATA PROCESSING         $15.00        $.00         $.00

 ADD    WIL-300177   WILKES, ASHLEY             ACCOUNTING               $8.35        $.00         $.00

 ADD    ZIP-714000   ZIPPER, ANN                CLERICAL                 $4.50        $.00         $.00


TOTAL TRANSACTIONS:    15      CORRECT:    15      ERRORS:    0     TOTAL GROSS PAY:    $1322.50     NET PAY:    $1021.65
```

SEQUENTIAL FILE UPDATING

UNIT XI

EXAMINATION

1. Match each term in Column B with an appropriate item in Column A.

Column A	Column B
____ a. Changes financial values	1. Posting transaction
____ b. Collective term for updating contents of a file	2. Addition transaction
____ c. Removes record from a file	3. Deletion transaction
____ d. Adds a new record to a file	4. File maintenance

2. List the four major processing steps of a sequential file update program in correct sequence.

3. Briefly explain the detailed programming specifications for a deletion transaction.

4. For each of the following processing conditions, state what type of transaction(s) is possible and why.

   a. Transaction ID low:

   b. Transaction ID equal:

   c. Transaction ID greater than:

True and False

T F 1. The module named "Obtain New Master Data" controls the transaction low, transaction equal, and transaction greater than processing.

T F 2. The driver module, Update Program Master, opens the files.

T F 3. The master file for this application is unblocked.

T F 4. The coded module, C000-PROCESS-EQUAL-RECORDS, processes changes, deletions, and error transactions only.

SEQUENTIAL FILE UPDATING

UNIT XI

ANSWERS TO EXAMINATION

1. a. 1
   b. 4
   c. 3
   d. 2

2. a. Obtain records
   b. Compare record IDs
   c. Update contents of master records
   d. Write update record

3. It must be code "D" and equal or else it is in error. Do not write to new file. Transaction only contains ID and code.

4. a. Transaction low - because both files are in sequence, the transaction must be an addition as there is no matching record on the master.

   b. Transaction equal - transaction matches master so must be change or deletion of this (existing) master

   c. Transaction greater than - all transactions for the compared master are done (or none existed) so now this master can be written out.

True and False

1. T
2. T
3. F
4. T

INDEXED SEQUENTIAL FILES I

UNIT XII

UNIT OBJECTIVE

After studying this and the following unit, the student should be able to apply elementary indexed sequential file access concepts and techniques in writing COBOL programs.

SPECIFIC OBJECTIVES

After studying this unit, students should be able to:

1. Match terms and definitions.
2. Explain meaning and usage of indexed sequential access files.
3. Recognize correct statements from the sample program's programming specifications.
4. List major functions from IPO and hierarchy charts for the sample program.
5. Recognize correct statements from the sample program's pseudocode specifications.
6. Recognize correct statements from the sample program's coding and listing.

INDEXED SEQUENTIAL FILES I

UNIT XII

SUGGESTED ACTIVITIES

I. Instructor

   A. Make transparencies; prepare lecture using this curriculum guide and supplemental materials.

   B. Provide students with objectives; discuss objectives.

   C. Make parallel assignments for reading and programming from the text(s).

   D. Provide students with information and assignment sheets; discuss these sheets.

   E. Provide students with additional supplementary information and exercises.

   F. Give examination.

II. Student

   A. Read objectives.

   B. Study information sheets and text assignments.

   C. Complete assignment sheets and supplementary exercises.

   D. Complete examination.

INSTRUCTIONAL MATERIALS

I. Unit Content

   A. Objectives

   B. Notes to the instructor

   C. Information sheets

   D. Handouts

      1. HO 1: Pseudocode for Get Valid Pair of Records Module

      2. HO 2: Pseudocode Master File for Write Customer Module

      3. HO 3: Source Listing of Sample Indexed Sequential Program (Parts 1, 2, 3, 4, and 5)

E. Transparency masters

1. TM 1: Format of Customer Master File
2. TM 2: Format of Input Records
3. TM 3: Format of the Exception Report
4. TM 4: Format of the Customer Master File Listing
5. TM 5: Overview of Sample Program to Load Indexed Sequential File
6. TM 6: IPO Chart - Load Customer Master File
7. TM 7: Hierarchy Chart for Sample Program
8. TM 8: Select Sentence - Indexed Sequential File
9. TM 9: Environment and Data Division Entries for Sequential Retrieval
10. TM 10: File Definition of Customer Master File

F. Assignment sheets

1. AS 1: Understanding the Sample Program Specifications

G. Answers to assignment sheets

H. Examination

I. Answers to examination

INDEXED SEQUENTIAL FILES I

UNIT XII

NOTES TO THE INSTRUCTOR

1. Job sheets for this unit are included in a comprehensive application in Unit XIII but some instructors may want their students to start now. If so, it is recommended that they start the file build and file print routines in this unit rather than where they would normally be done in Unit XIII.

2. JCL for indexed sequential files on IBM systems is complex. A sample JCL for the University of Missouri's IBM OS/MVS operating system is given but each computer system will have major differences. See your computer center personnel for help.

INDEXED SEQUENTIAL FILES I

UNIT XII

INFORMATION SHEETS

I. Terms and functional definitions

   A. **Random access** - involves going directly to specific records in a file for retrieval of their contents

   B. **Index** - pairs of record keys and their location in the file, used for random access

   C. **Indexed sequential** - allows both random and sequential access

   D. **Indexed sequential access method (ISAM)** - involves going directly to specific records in a file for retrieval of their contents

II. Introduction to indexed sequential (IS) access

   A. Indexed sequential means that a file can be accessed either randomly by means of an index or sequentially from beginning to end.

   B. Advantages - same file can support on-line, interactive applications and also batch processing of transactions, which allows economy and flexibility.

III. Sample IS program overview

   A. Sample program of this unit consists of the following:

      1. Loads (builds) a customer master file

      2. Customer master file is to be an IS file.

      3. After loading, a customer master list will be sequentially printed.

      4. An exception report will be printed during loading for any erroneous input records.

   B. Formats of the sample program:

      1. TM 1 shows the format of the Customer Master File, which is to be stored on disk as an indexed sequential file.

      2. TM 2 shows the format of the Input Records, which contain the contents necessary to load the Master File.

      3. TM 3 shows the format of the Exception Report for erroneous transactions.

      4. TM 4 shows the format of Customer Master File Listing, which prints the contents of the master records.

5. TM 5 shows an overview of the sample program to Load Indexed Sequential File.

C. Sample programming specifications are as follows:

A program is to be written to load a Customer Master File. The file is to be stored on a magnetic disk as an indexed sequential file. The format of the input records, the Customer Master File Record, the Customer Master File List, and the Exception Report are on transparencies in this unit. The program should be written to include the following processing.

1. The program should read a pair of records from the Customer Transaction File and create the Customer Master File. The Customer Master File is to be organized as an indexed sequential file.

2. All fields in the Customer Master Record will come directly from the data in the Customer Transaction Records.

3. The Customer Transaction File must be sorted by Transaction Code within Customer Number.

4. Before this information in the Transaction Records is used to build the Customer Master File, the data in the input records must be edited. If the data in the Transaction Records passes the editing criteria established in these specifications, the data can be used to create the Customer Master Record. If any of the editing tests is not met by the data in the Transaction Records, the input records for that customer must not be used for the Customer Record. Instead, the records should be reported on the Exception Report.

5. The following editing should be performed on the Transaction Records:

   a. The Transaction Code in the Transaction Records must be "1" or "2." If a record contains any other Transaction Code, it should not be used.

   b. There must be both a Type 1 and Type 2 input record for each customer. If both records are not present for each customer, an error message is to appear on an Exception Report; and the records are not to be placed in the Customer Master File. Further editing need not be performed.

   c. The Customer Number must be numeric.

   d. The Customer Name, Address, City, and State fields must not be blank.

   e. The State Code must be one of the valid 51 state codes (50 states and the District of Columbia).

   f. The Zip Code must be numeric.

   g. The Customer Discount field must be numeric and must not be greater than 20%.

h. The Last Activity Date must be numeric, and the month must not be less than 01 or greater than 12. The day of the month in the Transaction Record must not exceed the maximum days in that month.

i. The Customer Transaction Records must be in ascending sequence by Customer Number. If they are not, an entry should be made on the Exception Report.

j. There can be no duplicate Master Records. Therefore, if two pairs of Transaction Records with the same Customer Number are found, only one pair should be loaded. The error should be reported on the Exception Report.

6. All fields in each pair of Transaction Records should be checked if there is a valid pair of records.

7. If any field is found to be in error, the appropriate entry should be made on the Exception Report. The fields on the Exception Report are explained below.

   a. Customer Number - This field should contain the Customer Number of any record which contains one or more errors. It is to be group-indicated on the report; that is, if a single input record contains more than one error, the Customer Number should appear for the first error on the first line only.

   b. Field in Error - This field should identify the field which contains the error. If the error concerns the fact that there is a Type 1 record but no Type 2 record, or a Type 2 record but no Type 1 record, the Field in Error should be specified as "Customer Records."

   c. Type of Error - This field indicates the type of error which has been found in the input record.

   d. Format of the Report - The report headings and other formatting should be printed as illustrated on the accompanying printer spacing chart.

   e. The maximum number of detail lines on a single page of the Exception Report is forty (40) lines.

8. The Customer Master File list report should be printed after the Customer Master File has been completely created. The report should contain all of the data in the Customer Master File as illustrated on the accompanying printer spacing chart. The report format and headings should be printed as shown on the printer spacing chart. The maximum number of detail lines on a single page of the report is twenty (20) lines.

9. The program should be written in COBOL.

10. Note on the specifications and the input formats that the input records come on two cards. This was necessary as the data was larger than 80 columns.

11. There are two printed reports. Since they are not printed simultaneously, no special programming technique is needed; i.e., spooling is not necessary.

D. Decision Tables - your instructor may require the use of decision tables in your program. If you have not already done so, review decision tables in the tools unit of this guide or in your text.

IV. Sample IS program design - IPO and hierarchy charts

A. TM 6 gives the IPO chart for the top-level module.

B. TM 7 gives the detailed hierarchy chart.

V. Sample IS program design - pseudocode

A. The pseudocode for the module which obtains pairs of input records is in HO 1.

B. The pseudocode for the module which writes the customer master is in HO 2.

VI. Sample IS program design - coding and listing

A. Select statement for the IS file with sequential access is in TM 8.

B. The data division definition for the IS file is given in TM 9.

C. A sample set of coding for writing the IS file is given in HO 3.

NOTES:

1. Records must be in ascending sequence (because that is to be the order of the file).

2. No duplicate keys are allowed.

D. A sample set of coding for definition of the input file is given in TM 10. This definition is used for sequentially reading from the newly created master file.

NOTES:

1. This file is used for sequentially reading the newly created CUSTOMER MASTER FILE. There are other ways of reading which do not involve the definition of a separate file, but for now this is the simplest.

2. Notice particularly the assign statement of both input and output files to the same implementer name, MSTROUT. Although not

shown, the IBM JCL actually accomplishes this assignment so that both file names access the same physcial data set.

E. Source listing of sample IS program (HO 3)

HO 1

Pseudocode for Get Valid Pair of Records Module

Pseudocode Specifications

DATE: May 2	PROGRAMMER: Jane Austin	PROGRAM: Load Customer Master File
MODULE FUNCTION: Verify a Valid Pair of Records	REF: D010	MODULE NAME: Get Valid Pair of Recs

PSEUDOCODE / **REF:**

```
Set valid pair of records indicator to neutral
Read a record
IF there is a record
   Move record to first record work area
ENDIF
PERFORM UNTIL there is a valid pair of records or
         there are no more records
   Set valid first record indicator to neutral
   Set valid second record indicator to neutral
   Move record type to record 1 work area
   Move customer number to record 1 work area
   Verify valid first record                                  E000
   IF the first record is valid
      Read a record
      IF there is a record
         Move record to second record work area
         Move record type to record 2 work area
         Move customer number to record 2 work area
         Verify valid second record                           E010
         IF second record valid
            Indicate valid pair of records
         ELSE
            IF record type not = 1
               Read a record
               IF there is a record
                  Move record to first record work area
               ENDIF
            ELSE
               Move record from second record work area
                     to first record work area
            ENDIF
         ENDIF
      ELSE
         Move record 1 customer number to report work area
         Move "customer records" to report work area
         Move "type 1 but no type 2" to report work area
         Write exception report                               U000
      ENDIF
   ELSE
      Read a record
      IF there is a record
         Move the record to the first record work area
      ENDIF
   ENDIF
ENDPERFORM
```

FILES, RECORDS, FIELDS REQUIRED

- Input record area
- Valid pair of records indicator
- Valid first record indicator
- Valid second record indicator
- Work area - transaction 1
  - Customer number
  - Customer name
  - Customer address
  - Customer city
  - Customer state
  - Customer zip code
  - Transaction code
- Work area - transaction 2
  - Customer number
- Customer discount
  - Last activity date
  - Transaction code
- No more records indicator
- Work area to record type 1
- Record type (trans code)
  - Customer number
- There is a record indicator
- Work area for record type 2
  - Record type (trans code)
  - Customer number
- Exception report work area
  - Customer number
  - Field in error
  - Type of error
- "Type 1 but no type 2" constant

HO 2

Pseudocode Master File for Write Customer Module

Pseudocode Specifications

DATE: May 2	PROGRAMMER: Jane Austin	PROGRAM: Load Customer Master File
MODULE FUNCTION: Write the Customer Master File	REF: B010	MODULE NAME: Write Customer Master File

PSEUDOCODE

```
Move customer number to master area
Move customer name to master area
Move customer address to master area
Move customer city to master area
Move customer state to master area
Move customer zip code to master area
Move customer discount to master area
Move last activity date to master area
Write the customer master record
IF write was unsuccessful
   Move customer number to report work area
   Move "customer number" to report work area
   Move "duplicate or bad sequence" to report
      work area
   Write exception report
ENDIF
```

REF:

U000

FILES, RECORDS, FIELDS REQUIRED

- New customer master output area
  - Customer number
  - Customer name
  - Customer address
  - Customer city
  - Customer state
  - Customer zip code
  - Customer discount
  - Last activity date
- Transaction records data
  - Customer number
  - Customer name
  - Customer address
  - Customer city
  - Customer state
  - Customer zip code
  - Customer discount
  - Last activity date
- Report work area
  - Customer number
  - Field in error
  - Type of error
- "Customer number" constant
- "Duplicate or bad sequence" constant

Source Listing of Sample Indexed Sequential Program (Part 1 of 5)

```
 IDENTIFICATION DIVISION.

 PROGRAM-ID.    LOADCUST.
 AUTHOR.        SHELLY AND CASHMAN.
 INSTALLATION.  ANAHEIM.
 DATE-WRITTEN.  07/11/78.
 DATE-COMPILED. AUG 17,1978.
 SECURITY.      UNCLASSIFIED.

*****************************************************************
*                                                               *
*  THIS PROGRAM LOADS THE CUSTOMER MASTER FILE, WHICH IS AN     *
*  INDEXED SEQUENTIAL FILE. THE DATA FOR EACH CUSTOMER MASTER   *
*  RECORD IS STORED ON TWO TRANSACTION RECORDS WITH TRANSACTION*
*  CODE 1 AND 2. PRIOR TO BEING USED TO LOAD THE CUSTOMER       *
*  MASTER FILE THE TRANSACTION RECORDS MUST BE SORTED ON        *
*  TRANSACTION CODE WITHIN CUSTOMER NUMBER. AFTER BEING SORTED,*
*  THE TRANSACTION RECORDS ARE READ AND EDITED. THERE MUST BE   *
*  A VALID TYPE 1 AND VALID TYPE 2 TRANSACTION RECORD FOR EACH  *
*  MASTER RECORD WHICH IS LOADED. IF A TYPE 1 TRANSACTION IS    *
*  FOUND WITHOUT A TYPE 2, OR IF A TYPE 2 IS FOUND WITHOUT A    *
*  TYPE 1, OR IF ANY OTHER ERROR OCCURS, THE TRANSACTION        *
*  RECORDS ARE NOT USED FOR A MASTER RECORD. INSTEAD, THE       *
*  CUSTOMER NUMBER OF THE TRANSACTION IN ERROR, THE FIELD IN    *
*  ERROR AND THE TYPE OF ERROR ARE RECORDED ON AN EXCEPTION     *
*  REPORT. AFTER THE MASTER FILE IS CREATED, IT IS LISTED.      *
*                                                               *
*****************************************************************



 ENVIRONMENT DIVISION.

 CONFIGURATION SECTION.

 SOURCE-COMPUTER. IBM-370.
 OBJECT-COMPUTER. IBM-370.
 SPECIAL-NAMES.   C01 IS TO-THE-TOP-OF-THE-PAGE.

 INPUT-OUTPUT SECTION.

 FILE-CONTROL.
     SELECT CUSTOMER-TRANS-INPUT-FILE
         ASSIGN TO UR-S-SYSIN.
     SELECT CUSTOMER-MASTER-OUTPUT-FILE
         ASSIGN TO DA-I-MSTROUT
         ACCESS IS SEQUENTIAL
         RECORD KEY IS CUSTOMER-NUMBER-MSTROUT.
     SELECT PROGRAM-REPORT-FILE
         ASSIGN TO UR-S-SYSOUT.
     SELECT CUSTOMER-MASTER-INPUT-FILE
         ASSIGN TO DA-I-MSTROUT
         ACCESS IS SEQUENTIAL
         RECORD KEY IS CUSTOMER-NUMBER-MSTRIN.
     SELECT TRANSACTION-SORT-WORK-FILE
         ASSIGN TO DA-S-SORTWK01.
 DATA DIVISION.

 FILE SECTION.

 FD  CUSTOMER-TRANS-INPUT-FILE
       RECORD CONTAINS 80 CHARACTERS
       LABEL RECORDS ARE OMITTED
       DATA RECORD IS TRANSACTION-INPUT-RECORD.

 01  TRANSACTION-INPUT-RECORD          PIC X(80).

 FD  CUSTOMER-MASTER-OUTPUT-FILE
       BLOCK CONTAINS 20 RECORDS
       RECORD CONTAINS 88 CHARACTERS
       LABEL RECORDS ARE STANDARD
       DATA RECORD IS CUSTOMER-MASTER-OUTPUT-RECORD.

 01  CUSTOMER-MASTER-OUTPUT-RECORD.
     05  FILLER                        PIC X.
     05  CUSTOMER-NUMBER-MSTROUT       PIC X(5).
     05  CUSTOMER-NAME-MSTROUT         PIC X(20).
     05  CUSTOMER-ADDRESS-MSTROUT      PIC X(27).
     05  CUSTOMER-CITY-MSTROUT         PIC X(20).
     05  CUSTOMER-STATE-MSTROUT        PIC XX.
     05  CUSTOMER-ZIP-CODE-MSTROUT     PIC X(5).
     05  CUSTOMER-DISCOUNT-MSTROUT     PIC XX.
     05  LAST-ACTIVITY-DATE-MSTROUT    PIC X(6).

 FD  PROGRAM-REPORT-FILE
       RECORD CONTAINS 133 CHARACTERS
       LABEL RECORDS ARE OMITTED
       DATA RECORDS ARE EXCEPTION-REPORT-LINE,
                        MASTER-FILE-LIST-LINE.

 01  EXCEPTION-REPORT-LINE.
     05  CARRIAGE-CONTROL              PIC X.
     05  FILLER                        PIC X.
     05  CUSTOMER-NUMBER-EXCPRPT       PIC X(5).
     05  FILLER                        PIC X(6).
     05  FIELD-IN-ERROR-EXCPRPT        PIC X(20).
     05  FILLER                        PIC X(4).
     05  TYPE-OF-ERROR-EXCPRPT         PIC X(25).
     05  FILLER                        PIC X(71).

 01  MASTER-FILE-LIST-LINE.
     05  CARRIAGE-CONTROL              PIC X.
     05  FILLER                        PIC X.
     05  CUSTOMER-NUMBER-MSTRLIST      PIC X(5).
     05  FILLER                        PIC X(3).
     05  CUSTOMER-NAME-MSTRLIST        PIC X(20).
     05  FILLER                        PIC X(3).
     05  CUSTOMER-ADDRESS-MSTRLIST     PIC X(27).
     05  FILLER                        PIC X(3).
     05  CUSTOMER-CITY-MSTRLIST        PIC X(20).
     05  FILLER                        PIC X(3).
     05  CUSTOMER-STATE-MSTRLIST       PIC XX.
     05  FILLER                        PIC X(5).
     05  CUSTOMER-ZIP-CODE-MSTRLIST    PIC X(5).
     05  FILLER                        PIC X(3).
     05  CUSTOMER-DISCOUNT-MSTRLIST    PIC XX.
     05  PERCENT-SIGN-MSTRLIST         PIC X.
     05  FILLER                        PIC X(5).
     05  LAST-ACTIVITY-DATE-MSTRLIST PIC XXBXXBXX.
     05  FILLER                        PIC X(16).

 FD  CUSTOMER-MASTER-INPUT-FILE
       BLOCK CONTAINS 20 RECORDS
       RECORD CONTAINS 88 CHARACTERS
       LABEL RECORDS ARE STANDARD
       DATA RECORD IS CUSTOMER-MASTER-INPUT-RECORD.

 01  CUSTOMER-MASTER-INPUT-RECORD.
     05  FILLER                        PIC X.
     05  CUSTOMER-NUMBER-MSTRIN        PIC X(5).
     05  CUSTOMER-NAME-MSTRIN          PIC X(20).
     05  CUSTOMER-ADDRESS-MSTRIN       PIC X(27).
     05  CUSTOMER-CITY-MSTRIN          PIC X(20).
     05  CUSTOMER-STATE-MSTRIN         PIC XX.
     05  CUSTOMER-ZIP-CODE-MSTRIN      PIC X(5).
     05  CUSTOMER-DISCOUNT-MSTRIN      PIC XX.
     05  LAST-ACTIVITY-DATE-MSTRIN     PIC X(6).

 SD  TRANSACTION-SORT-WORK-FILE
       RECORD CONTAINS 80 CHARACTERS
       DATA RECORD IS TRANSACTION-SORT-WORK-RECORD.

 01  TRANSACTION-SORT-WORK-RECORD.
     05  CUSTOMER-NUMBER-SORTREC       PIC X(5).
     05 FILLER                         PIC X(74).
     05  TRANSACTION-CODE-SORTREC      PIC X.

 WORKING-STORAGE SECTION.

 01  PROGRAM-INDICATORS.
     05  ARE-THERE-MORE-TRAN-RECORDS       PIC XXX VALUE 'YES'.
         88  THERE-ARE-MORE-TRAN-RECORDS           VALUE 'YES'.
         88  THERE-ARE-NO-MORE-TRAN-RECORDS        VALUE 'NO '.
     05  ARE-TRAN-RECORDS-VALID            PIC XXX.
         88  TRAN-RECORDS-ARE-VALID                VALUE 'YES'.
         88  TRAN-RECORDS-ARE-NOT-VALID            VALUE 'NO '.
     05  ARE-THERE-MORE-MSTR-RECORDS       PIC XXX VALUE 'YES'.
         88  THERE-ARE-MORE-MSTR-RECORDS           VALUE 'YES'.
         88  THERE-ARE-NO-MORE-MSTR-RECORDS        VALUE 'NO '.
     05  ARE-THE-PAIR-OF-RECORDS-VALID   PIC XXX.
         88  THE-PAIR-OF-RECORDS-ARE-VALID         VALUE 'YES'.
     05  IS-THE-FIRST-RECORD-VALID         PIC XXX.
         88  THE-FIRST-RECORD-IS-VALID             VALUE 'YES'.
         88  THE-FIRST-RECORD-IS-NOT-VALID         VALUE 'NO '.
     05  IS-THE-SECOND-RECORD-VALID        PIC XXX.
         88  THE-SECOND-RECORD-IS-VALID            VALUE 'YES'.
         88  THE-SECOND-RECORD-IS-NOT-VALID        VALUE 'NO '.

 01  PROGRAM-COMPARE-AREAS.
     05  PREVIOUS-CUSTOMER-NUMBER      PIC X(5)     VALUE LOW-VALUES.

 01  PROGRAM-WORK-AREAS.
     05  EXCEPTION-REPORT-WORK-AREA.
         10  CUSTOMER-NUMBER-EXCP-WORK         PIC X(5).
         10  FIELD-IN-ERROR-EXCP-WORK          PIC X(20).
         10  TYPE-OF-ERROR-EXCP-WORK           PIC X(25).
     05  TYPE-1-TRANS-WORK-AREA.
         10  TRANSACTION-CODE-TRANS1-WORK      PIC X.
         10  CUSTOMER-NUMBER-TRANS1-WORK       PIC X(5).
     05  TYPE-2-TRANS-WORK-AREA.
         10  TRANSACTION-CODE-TRANS2-WORK      PIC X.
         10  CUSTOMER-NUMBER-TRANS2-WORK       PIC X[illegible]
     05  CUSTOMER-NUMBER-WORK.
         10  FIRST-FOUR-CHAR-CUST-NUMB-WORK  PIC X(4).
         10  FIFTH-CHAR-CUST-NUMB-WORK         PIC X.
             88  VALID-CHECK-DIGIT-FOR-TEN      VALUE 'X'.

 01  MOD-11-PASS-AREA-WORK.
         10  FIELD-TO-BE-CHECKED-MOD11-WORK  PIC X(20).
         10  DID-CUST-NUMBER-PASS-MOD11        PIC XXX.
             88  CUST-NUMBER-PASSED-MOD11           VALUE 'YES'.
             88  CUST-NUMBER-DID-NOT-PASS-MOD11  VALUE 'NO '.

 01  PROGRAM-CONSTANTS.
     05  CUSTOMER-NAME-FIELD-MSG                 PIC X(20) VALUE
                                               'CUSTOMER NAME        '.
     05  BLANK-FIELD-ERROR-MSG                   PIC X(25) VALUE
                                          'FIELD CONTAINS SPACES      '.
     05  CUSTOMER-ADDRESS-FIELD-MSG              PIC X(20) VALUE
                                               'CUSTOMER ADDRESS     '.
     05  CUSTOMER-CITY-FIELD-MSG                 PIC X(20) VALUE
                                               'CUSTOMER CITY        '.
     05  CUSTOMER-STATE-FIELD-MSG                PIC X(20) VALUE
                                               'CUSTOMER STATE       '.
     05  BAD-STATE-CODE-ERROR-MSG                PIC X(25) VALUE
                                          'INVALID STATE CODE         '.
     05  CUSTOMER-ZIP-CODE-FIELD-MSG             PIC X(20) VALUE
                                               'CUSTOMER ZIP CODE    '.
     05  FIELD-NOT-NUMERIC-ERROR-MSG             PIC X(25) VALUE
                                          'FIELD IS NOT NUMERIC       '.
     05  CUSTOMER-DISCOUNT-FIELD-MSG             PIC X(20) VALUE
                                               'CUSTOMER DISCOUNT    '.
```

BEST COPY AVAILABLE

HO 3 (cont.)

Source Listing (Part 2 of 5)

```
012050     05  MAXIMUM-DISCOUNT-PERCENT               PIC XX      VALUE '20'.
012060     05  BAD-DISCOUNT-ERROR-MSG                 PIC X(25) VALUE
012070                                     'DISCOUNT EXCEEDS MAXIMUM '.
012080     05  LAST-ACTIVITY-DATE-FIELD-MSG           PIC X(20) VALUE
012090                                          'LAST ACTIVITY DATE  '.
012120     05  BAD-MONTH-ERROR-MSG                    PIC X(25) VALUE
012130                                     'INVALID MONTH            '.
012140     05  FIRST-DAY-OF-MONTH-NUMBER              PIC XX      VALUE '01'.
012150     05  BAD-DAY-ERROR-MSG                      PIC X(25) VALUE
012160                                     'INVALID DAY              '.
012170     05  CUSTOMER-RECORDS-FIELD-MSG             PIC X(20) VALUE
012180                                          'CUSTOMER RECORDS    '.
012190     05  TYPE1-BUT-NO-TYPE2-ERROR-MSG           PIC X(25) VALUE
012200                                     'TYPE 1 BUT NO TYPE 2     '.
012210     05  TRANSACTION-TYPE-ONE                   PIC X       VALUE '1'.
012220     05  TRANSACTION-TYPE-TWO                   PIC X       VALUE '2'.
013010     05  TYPE2-BUT-NO-TYPE1-ERROR-MSG           PIC X(25) VALUE
013020                                     'TYPE 2 BUT NO TYPE 1     '.
013030     05  BAD-TRANS-CODE-ERROR-MSG               PIC X(25) VALUE
013040                                     'INVALID RECORD TYPE      '.
013050     05  CUSTOMER-NUMBER-FIELD-MSG              PIC X(20) VALUE
013060                                          'CUSTOMER NUMBER     '.
013070     05  FAILS-MOD11-ERROR-MSG                  PIC X(25) VALUE
013080                                     'FAILS MOD-11 CHECK       '.
013090     05  INVALID-WRITE-MSG                      PIC X(25) VALUE
013100                                     'DUPLICATE OR BAD SEQUENCE'.
013110
013120 01  PROGRAM-TABLES.
013130     05  STATE-TABLE.
013140         10  VALID-STATE-CONSTANTS.
013150             15  FILLER             PIC XX        VALUE 'AK'.
013160             15  FILLER             PIC XX        VALUE 'AL'.
013170             15  FILLER             PIC XX        VALUE 'AR'.
013180             15  FILLER             PIC XX        VALUE 'AZ'.
013190             15  FILLER             PIC XX        VALUE 'CA'.
013200             15  FILLER             PIC XX        VALUE 'CO'.
013210             15  FILLER             PIC XX        VALUE 'CT'.
013220             15  FILLER             PIC XX        VALUE 'DC'.
013230             15  FILLER             PIC XX        VALUE 'DE'.
014010             15  FILLER             PIC XX        VALUE 'FL'.
014020             15  FILLER             PIC XX        VALUE 'GA'.
014030             15  FILLER             PIC XX        VALUE 'HI'.
014040             15  FILLER             PIC XX        VALUE 'IA'.
014050             15  FILLER             PIC XX        VALUE 'ID'.
014060             15  FILLER             PIC XX        VALUE 'IL'.
014070             15  FILLER             PIC XX        VALUE 'IN'.
014080             15  FILLER             PIC XX        VALUE 'KS'.
014090             15  FILLER             PIC XX        VALUE 'KY'.
014100             15  FILLER             PIC XX        VALUE 'LA'.
014110             15  FILLER             PIC XX        VALUE 'MA'.
014120             15  FILLER             PIC XX        VALUE 'MD'.
014130             15  FILLER             PIC XX        VALUE 'ME'.
014140             15  FILLER             PIC XX        VALUE 'MI'.
014150             15  FILLER             PIC XX        VALUE 'MN'.
014160             15  FILLER             PIC XX        VALUE 'MO'.
014170             15  FILLER             PIC XX        VALUE 'MS'.
014180             15  FILLER             PIC XX        VALUE 'MT'.
014190             15  FILLER             PIC XX        VALUE 'NB'.
014200             15  FILLER             PIC XX        VALUE 'NC'.
015010             15  FILLER             PIC XX        VALUE 'ND'.
015020             15  FILLER             PIC XX        VALUE 'NH'.
015030             15  FILLER             PIC XX        VALUE 'NJ'.
015040             15  FILLER             PIC XX        VALUE 'NM'.
015050             15  FILLER             PIC XX        VALUE 'NV'.
015060             15  FILLER             PIC XX        VALUE 'NY'.
015070             15  FILLER             PIC XX        VALUE 'OH'.
015080             15  FILLER             PIC XX        VALUE 'OK'.
015090             15  FILLER             PIC XX        VALUE 'OR'.
015100             15  FILLER             PIC XX        VALUE 'PA'.
015110             15  FILLER             PIC XX        VALUE 'RI'.
015120             15  FILLER             PIC XX        VALUE 'SC'.
015130             15  FILLER             PIC XX        VALUE 'SD'.
015140             15  FILLER             PIC XX        VALUE 'TN'.
015150             15  FILLER             PIC XX        VALUE 'TX'.
015160             15  FILLER             PIC XX        VALUE 'UT'.
015170             15  FILLER             PIC XX        VALUE 'VA'.
015180             15  FILLER             PIC XX        VALUE 'VT'.
015190             15  FILLER             PIC XX        VALUE 'WA'.
015200             15  FILLER             PIC XX        VALUE 'WI'.
016010             15  FILLER             PIC XX        VALUE 'WV'.
016020             15  FILLER             PIC XX        VALUE 'WY'.
016030         10  VALID-STATE-TABLE REDEFINES VALID-STATE-CONSTANTS
016040                                          OCCURS 51 TIMES
016050                                          ASCENDING KEY IS
016060                                             STATE-CODE
016070                                          INDEXED BY
016080                                             STATE-CODE-INDEX.
016090             15  STATE-CODE         PIC XX.
016100     05  MONTH-DAY-TABLE.
016110         10  MONTH-DAY-CONSTANTS.
016120             15  FILLER             PIC X(4)      VALUE '0131'.
016130             15  FILLER             PIC X(4)      VALUE '0229'.
016140             15  FILLER             PIC X(4)      VALUE '0331'.
016150             15  FILLER             PIC X(4)      VALUE '0430'.
016160             15  FILLER             PIC X(4)      VALUE '0531'.
016170             15  FILLER             PIC X(4)      VALUE '0630'.
016180             15  FILLER             PIC X(4)      VALUE '0731'.
016190             15  FILLER             PIC X(4)      VALUE '0831'.
016200             15  FILLER             PIC X(4)      VALUE '0930'.
017010             15  FILLER             PIC X(4)      VALUE '1031'.
017020             15  FILLER             PIC X(4)      VALUE '1130'.
017030             15  FILLER             PIC X(4)      VALUE '1231'.
017040         10  MONTHS-AND-DAYS-TABLE REDEFINES MONTH-DAY-CONSTANTS
017050                                          OCCURS 12 TIMES
017060                                          INDEXED BY
017070                                             MONTH-DAY-INDEX.
```

```
017080             15  MONTH-TABLE        PIC XX.
017090             15  NUMBER-OF-DAYS-IN-MONTH PIC XX.
017100
017110 01  INPUT-RECORD-WORK-AREAS.
017120     05  SORTED-TRANS-RECORD-TYPE-1.
017130         10  CUSTOMER-NUMBER-TRANSORT1                    PIC X(5).
017140         10  CUSTOMER-NAME-TRANSORT1                      PIC X(20).
017150         10  CUSTOMER-ADDRESS-TRANSORT1                   PIC X(27).
017160         10  CUSTOMER-CITY-TRANSORT1                      PIC X(20).
017170         10 CUSTOMER-STATE-TRANSORT1                      PIC XX.
017180         10 CUSTOMER-ZIP-CODE-TRANSORT1                   PIC X(5).
017190         10  TRANSACTION-CODE-TRANSORT1                   PIC X.
017200     05  SORTED-TRANS-RECORD-TYPE-2.
017210         10  CUSTOMER-NUMBER-TRANSORT2                    PIC X(5).
017220         10  CUSTOMER-DISCOUNT-TRANSORT2                  PIC XX.
017230         10  LAST-ACTIVITY-DATE-TRANSORT2.
017240             15  MONTH-LAST-ACTIVITY-TRANSORT2            PIC 99.
017250                 88  VALID-MONTH                          VALUE 01 THRU 12.
017260             15  DAY-LAST-ACTIVITY-TRANSORT2              PIC XX.
017270             15  YEAR-LAST-ACTIVITY-TRANSORT2             PIC XX.
017280         10  FILLER                                       PIC X(66).
017290         10  TRANSACTION-CODE-TRANSORT2                   PIC X.
017300
017310 01  PRINTER-CONTROL-EXCEPTION-RPT.
017320     05  PROPER-SPACING-EXCP-RPT        PIC 9.
017330     05  SPACE-ONE-LINE-EXCP-RPT        PIC 9    VALUE 1.
017340     05  SPACE-TWO-LINES-EXCP-RPT       PIC 9    VALUE 2.
017350     05  LINES-PRINTED-EXCP-RPT         PIC S999 VALUE ZERO
017360                                                 USAGE IS COMP-3.
017370     05  PAGE-SIZE-EXCP-RPT             PIC 999  VALUE 40
017380                                                 USAGE IS COMP-3.
017390     05  PAGE-NUMBER-EXCP-RPT           PIC S999 VALUE +1
017400                                                 USAGE IS COMP-3.
018010         88  FIRST-PAGE-EXCP-RPT                 VALUE +1.
018020
018030 01  PRINTER-CONTROL-MASTER-LIST.
018040     05  PROPER-SPACING-MSTR-LIST       PIC 9.
018050     05  SPACE-ONE-LINE-MSTR-LIST       PIC 9    VALUE 1.
018060     05  SPACE-TWO-LINES-MSTR-LIST      PIC 9    VALUE 2.
018070     05  LINES-PRINTED-MSTR-LIST        PIC S999 VALUE ZERO
018080                                                 USAGE IS COMP-3.
018090     05  PAGE-SIZE-MSTR-LIST            PIC 999  VALUE 20
018100                                                 USAGE IS COMP-3.
018110     05  PAGE-NUMBER-MSTR-LIST          PIC S999 VALUE +1
018120                                                 USAGE IS COMP-3.
018130         88  FIRST-PAGE-MSTR-LIST                VALUE +1.
018140
018150 01  EXCEPTION-REPORT-HEADINGS.
018160     05  FIRST-HEADING-LINE-EXCP-RPT.
018170         10  CARRIAGE-CONTROL     PIC X.
018180         10  DATE-HDG1-EXCP-RPT   PIC X(8).
018190         10  FILLER               PIC X(3)     VALUE SPACES.
018200         10  FILLER               PIC X(9)     VALUE 'CUSTOMER '.
019010         10  FILLER               PIC X(7)     VALUE 'MASTER '.
019020         10  FILLER               PIC X(5)     VALUE 'LOAD '.
019030         10  FILLER               PIC X(10)    VALUE 'EXCEPTION '.
019040         10  FILLER               PIC X(6)     VALUE 'REPORT'.
019050         10  FILLER               PIC X(4)     VALUE SPACES.
019060         10  FILLER               PIC X(5)     VALUE 'PAGE '.
019070         10  PAGE-NUMBER-HDG1-EXCP-RPT PIC ZZ9.
019080         10  FILLER               PIC X(72).
019090     05  SECOND-HEADING-LINE-EXCP-RPT.
019100         10  CARRIAGE-CONTROL     PIC X.
019110         10  FILLER               PIC X(8)     VALUE 'CUSTOMER'.
019120         10  FILLER               PIC X(11)    VALUE SPACES.
019130         10  FILLER               PIC X(5)     VALUE 'FIELD'.
019140         10  FILLER               PIC X(108)   VALUE SPACES.
019150     05  THIRD-HEADING-LINE-EXCP-RPT.
019160         10  CARRIAGE-CONTROL     PIC X.
019170         10  FILLER               PIC X        VALUE SPACE.
019180         10  FILLER               PIC X(6)     VALUE 'NUMBER'.
019190         10  FILLER               PIC X(10)    VALUE SPACES.
019200         10  FILLER               PIC X(8)     VALUE 'IN ERROR'.
020010         10  FILLER               PIC X(16)    VALUE SPACES.
020020         10  FILLER               PIC X(13)    VALUE
020030                                             'TYPE OF ERROR'.
020040         10  FILLER               PIC X(78)    VALUE SPACES.
020050
020060 01  MASTER-LIST-REPORT-HEADINGS.
020070     05  FIRST-HEADING-LINE-MSTR-LIST.
020080         10  CARRIAGE-CONTROL     PIC X.
020090         10  FILLER               PIC X(5)     VALUE 'DATE '.
020100         10  DATE-HDG1-MSTR-LIST  PIC X(8).
020110         10  FILLER               PIC X(32)    VALUE SPACES.
020120         10  FILLER               PIC X(9)     VALUE 'CUSTOMER '.
020130         10  FILLER               PIC X(7)     VALUE 'MASTER '.
020140         10  FILLER               PIC X(5)     VALUE 'FILE'.
020150         10  FILLER               PIC X(4)     VALUE 'LIST'.
020160         10  FILLER               PIC X(38)    VALUE SPACES.
020170         10  FILLER               PIC X(5)     VALUE 'PAGE '.
020180         10  PAGE-NUMBER-HDG1-MSTR-LIST PIC ZZ9.
020190         10  FILLER               PIC X(16)    VALUE SPACES.
020200     05  SECOND-HEADING-LINE-MSTR-LIST.
021010         10  CARRIAGE-CONTROL     PIC X.
021020         10  FILLER               PIC X(8)     VALUE 'CUSTOMER'.
021030         10  FILLER               PIC X(7)     VALUE SPACES.
021040         10  FILLER               PIC X(8)     VALUE 'CUSTOMER'.
021050         10  FILLER               PIC X(10)    VALUE SPACES.
021060         10  FILLER               PIC X(8)     VALUE 'CUSTOMER'.
021070         10  FILLER               PIC X(19)    VALUE SPACES.
021080         10  FILLER               PIC X(8)     VALUE 'CUSTOMER'.
021090         10  FILLER               PIC X(6)     VALUE SPACES.
021100         10  FILLER               PIC X(8)     VALUE 'CUSTOMER'.
021110         10  FILLER               PIC X(3)     VALUE SPACES.
021120         10  FILLER               PIC X(3)     VALUE 'ZIP'.
021130         10  FILLER               PIC X(14)    VALUE SPACES.
```

BEST COPY AVAILABLE

Source Listing (Part 3 of 5)

```
           10  FILLER              PIC X(4)     VALUE 'LAST'.
           10  FILLER              PIC X(18)    VALUE SPACES.
       05  THIRD-HEADING-LINE-MSTR-LIST.
           10  CARRIAGE-CONTROL    PIC X.
           10  FILLER              PIC X        VALUE SPACE.
           10  FILLER              PIC X(6)     VALUE 'NUMBER'.
           10  FILLER              PIC X(10)    VALUE SPACES.
           10  FILLER              PIC X(4)     VALUE 'NAME'.
           10  FILLER              PIC X(20)    VALUE SPACES.
           10  FILLER              PIC X(7)     VALUE 'ADDRESS'.
           10  FILLER              PIC X(22)    VALUE SPACES.
           10  FILLER              PIC X(4)     VALUE 'CITY'.
           10  FILLER              PIC X(10)    VALUE SPACES.
           10  FILLER              PIC X(5)     VALUE 'STATE'.
           10  FILLER              PIC X(3)     VALUE SPACES.
           10  FILLER              PIC X(4)     VALUE 'CODE'.
           10  FILLER              PIC XX       VALUE SPACES.
           10  FILLER              PIC X(8)     VALUE 'DISCOUNT'.
           10  FILLER              PIC X(2)     VALUE SPACES.
           10  FILLER              PIC X(8)     VALUE 'ACTIVITY'.
           10  FILLER              PIC X(10)    VALUE SPACES.
       PROCEDURE DIVISION.

      *****************************************************************
      *                                                               *
      *  THE FUNCTION OF THIS MODULE IS TO LOAD THE CUSTOMER MASTER   *
      *  FILE. IT IS ENTERED FROM AND EXITS TO THE OPERATING SYSTEM.  *
      *                                                               *
      *****************************************************************

       BEGINNING-SECTION SECTION.

       A000-LOAD-CUSTOMER-MASTER-FILE.

           OPEN OUTPUT PROGRAM-REPORT-FILE
                       CUSTOMER-MASTER-OUTPUT-FILE.
           SORT TRANSACTION-SORT-WORK-FILE
                ON ASCENDING KEY CUSTOMER-NUMBER-SORTREC
                                 TRANSACTION-CODE-SORTREC
                USING CUSTOMER-TRANS-INPUT-FILE
                OUTPUT PROCEDURE A001-SORT-OUTPUT-PROCEDURE.
           CLOSE CUSTOMER-MASTER-OUTPUT-FILE.
           OPEN INPUT  CUSTOMER-MASTER-INPUT-FILE.
           PERFORM A000-LIST-CUSTOMER-MASTER.
           CLOSE PROGRAM-REPORT-FILE
                 CUSTOMER-MASTER-INPUT-FILE.
           STOP RUN.



      *****************************************************************
      *                                                               *
      *  THIS SECTION IS ENTERED TO PROCESS THE SORTED TRANSACTION    *
      *  RECORDS. IT IS ENTERED FROM THE SORT STATEMENT IN THE        *
      *  A000-LOAD-CUSTOMER-MASTER-FILE MODULE AND EXITS BACK TO THE  *
      *  STATEMENT FOLLOWING THE SORT VERB THROUGH THE USE OF A       *
      *  GO TO STATEMENT WHICH TRANSFERS CONTROL TO THE END OF THIS   *
      *  SECTION.                                                     *
      *                                                               *
      *****************************************************************

       A001-SORT-OUTPUT-PROCEDURE SECTION.

       A001-PROCESS-RECORDS.

           PERFORM B000-OBTAIN-VALID-LOAD-DATA.
           PERFORM A002-WRITE-AND-READ
               UNTIL THERE-ARE-NO-MORE-TRAN-RECORDS.
           GO TO A001-SORT-PROCEDURE-EXIT.



       A002-WRITE-AND-READ.

           PERFORM W010-WRITE-CUSTOMER-MSTR-FILE.
           PERFORM B000-OBTAIN-VALID-LOAD-DATA.
      *****************************************************************
      *                                                               *
      *  THIS MODULE IS ENTERED TO OBTAIN VALID MASTER LOAD DATA.     *
      *  IT IS ENTERED FROM AND EXITS TO THE A000-LOAD-CUSTOMER-      *
      *  MASTER-FILE MODULE.                                          *
      *                                                               *
      *****************************************************************

       B000-OBTAIN-VALID-LOAD-DATA.

           MOVE SPACES TO ARE-TRAN-RECORDS-VALID.
           PERFORM C000-OBTAIN-PAIR-OF-RECORDS.
           PERFORM B001-EDIT-AND-READ
               UNTIL TRAN-RECORDS-ARE-VALID OR
                     THERE-ARE-NO-MORE-TRAN-RECORDS.



       B001-EDIT-AND-READ.

           PERFORM C010-EDIT-TYPE-1-RECORD.
           PERFORM C020-EDIT-TYPE-2-RECORD.
           IF TRAN-RECORDS-ARE-NOT-VALID
              PERFORM C000-OBTAIN-PAIR-OF-RECORDS
              MOVE SPACES TO ARE-TRAN-RECORDS-VALID
           ELSE
              MOVE 'YES' TO ARE-TRAN-RECORDS-VALID.
      *****************************************************************
      *                                                               *
      *  THIS MODULE WRITES THE CUSTOMER MASTER FILE. IT IS ENTERED   *
      *  FROM AND EXITS TO THE A000-LOAD-CUSTOMER-MASTER-FILE MODULE.*
      *                                                               *
      *****************************************************************

       W010-WRITE-CUSTOMER-MSTR-FILE.

           MOVE CUSTOMER-NUMBER-TRANSORT1 TO CUSTOMER-NUMBER-MSTROUT.
           MOVE CUSTOMER-NAME-TRANSORT1 TO CUSTOMER-NAME-MSTROUT.
           MOVE CUSTOMER-ADDRESS-TRANSORT1 TO CUSTOMER-ADDRESS-MSTROUT.
           MOVE CUSTOMER-CITY-TRANSORT1 TO CUSTOMER-CITY-MSTROUT.
           MOVE CUSTOMER-STATE-TRANSORT1 TO CUSTOMER-STATE-MSTROUT.
           MOVE CUSTOMER-ZIP-CODE-TRANSORT1 TO
                   CUSTOMER-ZIP-CODE-MSTROUT.
           MOVE CUSTOMER-DISCOUNT-TRANSORT2 TO
                   CUSTOMER-DISCOUNT-MSTROUT.
           MOVE LAST-ACTIVITY-DATE-TRANSORT2 TO
                   LAST-ACTIVITY-DATE-MSTROUT.
           WRITE CUSTOMER-MASTER-OUTPUT-RECORD
                   INVALID KEY
                      MOVE CUSTOMER-NUMBER-TRANSORT1 TO
                              CUSTOMER-NUMBER-EXCP-WORK
                      MOVE CUSTOMER-NUMBER-FIELD-MSG TO
                              CUSTOMER-NUMBER-EXCP-WORK
                      MOVE INVALID-WRITE-MSG TO TYPE-OF-ERROR-EXCP-WORK
                      PERFORM U000-WRITE-EXCEPTION-REPORT.
      *****************************************************************
      *                                                               *
      *  THIS MODULE OBTAINS A VALID PAIR OF RECORDS. IT IS ENTERED   *
      *  FROM AND EXITS TO THE B000-OBTAIN-VALID-LOAD-DATA MODULE.    *
      *                                                               *
      *****************************************************************

       C000-OBTAIN-PAIR-OF-RECORDS.

           MOVE SPACES TO ARE-THE-PAIR-OF-RECORDS-VALID.
           PERFORM C002-READ-A-RECORD.
           IF THERE-ARE-MORE-TRAN-RECORDS
               MOVE TRANSACTION-SORT-WORK-RECORD TO
                       SORTED-TRANS-RECORD-TYPE-1.
           PERFORM C001-FIND-VALID-PAIR
                   UNTIL THE-PAIR-OF-RECORDS-ARE-VALID OR
                         THERE-ARE-NO-MORE-TRAN-RECORDS.



       C001-FIND-VALID-PAIR.

           MOVE SPACES TO IS-THE-FIRST-RECORD-VALID.
           MOVE SPACES TO IS-THE-SECOND-RECORD-VALID.
           MOVE TRANSACTION-CODE-TRANSORT1 TO
                   TRANSACTION-CODE-TRANS1-WORK.
           MOVE CUSTOMER-NUMBER-TRANSORT1 TO
                   CUSTOMER-NUMBER-TRANS1-WORK.
           PERFORM D000-VERIFY-VALID-FIRST-REC.
           IF THE-FIRST-RECORD-IS-VALID
              PERFORM C002-READ-A-RECORD
              IF THERE-ARE-MORE-TRAN-RECORDS
                  MOVE TRANSACTION-SORT-WORK-RECORD TO
                          SORTED-TRANS-RECORD-TYPE-2
                  MOVE TRANSACTION-CODE-TRANSORT2 TO
                          TRANSACTION-CODE-TRANS2-WORK
                  MOVE CUSTOMER-NUMBER-TRANSORT2 TO
                          CUSTOMER-NUMBER-TRANS2-WORK
                  PERFORM D010-VERIFY-VALID-SECOND-REC
                  IF THE-SECOND-RECORD-IS-VALID
                     MOVE 'YES' TO ARE-THE-PAIR-OF-RECORDS-VALID
                  ELSE
                     IF TRANSACTION-CODE-TRANSORT2 NOT =
                             TRANSACTION-TYPE-ONE
                         PERFORM C002-READ-A-RECORD
                         IF THERE-ARE-MORE-TRAN-RECORDS
                            MOVE TRANSACTION-SORT-WORK-RECORD TO
                                    SORTED-TRANS-RECORD-TYPE-1
                         ELSE
                            NEXT SENTENCE
                     ELSE
                         MOVE SORTED-TRANS-RECORD-TYPE-2 TO
                                 SORTED-TRANS-RECORD-TYPE-1
              ELSE
                  MOVE CUSTOMER-NUMBER-TRANS1-WORK TO
                          CUSTOMER-NUMBER-EXCP-WORK
                  MOVE CUSTOMER-RECORDS-FIELD-MSG TO
                          FIELD-IN-ERROR-EXCP-WORK
                  MOVE TYPE1-BUT-NO-TYPE2-ERROR-MSG TO
                          TYPE-OF-ERROR-EXCP-WORK
                  PERFORM U000-WRITE-EXCEPTION-REPORT
           ELSE
              PERFORM C002-READ-A-RECORD
              IF THERE-ARE-MORE-TRAN-RECORDS
                  MOVE TRANSACTION-SORT-WORK-RECORD TO
                          SORTED-TRANS-RECORD-TYPE-1.



       C002-READ-A-RECORD.

           RETURN TRANSACTION-SORT-WORK-FILE
               AT END
                   MOVE 'NO ' TO ARE-THERE-MORE-TRAN-RECORDS.
```

BEST COPY AVAILABLE

Source Listing (Part 4 of 5)

```
****************************************************************
*                                                              *
*  THE FUNCTION OF THIS MODULE IS TO EDIT A TYPE 1 TRANSACTION *
*  RECORD. IT IS ENTERED FROM AND EXITS TO THE                 *
*  B000-OBTAIN-VALID-LOAD-DATA MODULE.                         *
*                                                              *
****************************************************************

 C010-EDIT-TYPE-1-RECORD.

     MOVE CUSTOMER-NUMBER-TRANSORT1 TO CUSTOMER-NUMBER-WORK.
     PERFORM U020-EDIT-CUSTOMER-NUMBER.
     IF CUSTOMER-NAME-TRANSORT1 = SPACES
        MOVE CUSTOMER-NUMBER-TRANSORT1 TO
              CUSTOMER-NUMBER-EXCP-WORK
        MOVE CUSTOMER-NAME-FIELD-MSG TO
              FIELD-IN-ERROR-EXCP-WORK
        MOVE BLANK-FIELD-ERROR-MSG TO TYPE-OF-ERROR-EXCP-WORK
        PERFORM U000-WRITE-EXCEPTION-REPORT
        MOVE 'NO ' TO ARE-TRAN-RECORDS-VALID.
     IF CUSTOMER-ADDRESS-TRANSORT1 = SPACES
        MOVE CUSTOMER-NUMBER-TRANSORT1 TO
              CUSTOMER-NUMBER-EXCP-WORK
        MOVE CUSTOMER-ADDRESS-FIELD-MSG TO
              FIELD-IN-ERROR-EXCP-WORK
        MOVE BLANK-FIELD-ERROR-MSG TO TYPE-OF-ERROR-EXCP-WORK
        PERFORM U000-WRITE-EXCEPTION-REPORT
        MOVE 'NO ' TO ARE-TRAN-RECORDS-VALID.
     IF CUSTOMER-CITY-TRANSORT1 = SPACES
        MOVE CUSTOMER-NUMBER-TRANSORT1 TO
              CUSTOMER-NUMBER-EXCP-WORK
        MOVE CUSTOMER-CITY-FIELD-MSG TO
              FIELD-IN-ERROR-EXCP-WORK
        MOVE BLANK-FIELD-ERROR-MSG TO TYPE-OF-ERROR-EXCP-WORK
        PERFORM U000-WRITE-EXCEPTION-REPORT
        MOVE 'NO ' TO ARE-TRAN-RECORDS-VALID.
     SEARCH ALL VALID-STATE-TABLE
           AT END
              MOVE CUSTOMER-NUMBER-TRANSORT1 TO
                    CUSTOMER-NUMBER-EXCP-WORK
              MOVE CUSTOMER-STATE-FIELD-MSG TO
                    FIELD-IN-ERROR-EXCP-WORK
              MOVE BAD-STATE-CODE-ERROR-MSG TO
                    TYPE-OF-ERROR-EXCP-WORK
              PERFORM U000-WRITE-EXCEPTION-REPORT
              MOVE 'NO ' TO ARE-TRAN-RECORDS-VALID
           WHEN STATE-CODE (STATE-CODE-INDEX) =
                 CUSTOMER-STATE-TRANSORT1
              NEXT SENTENCE.
     IF CUSTOMER-ZIP-CODE-TRANSORT1 IS NOT NUMERIC
        MOVE CUSTOMER-NUMBER-TRANSORT1 TO
              CUSTOMER-NUMBER-EXCP-WORK
        MOVE CUSTOMER-ZIP-CODE-FIELD-MSG TO
              FIELD-IN-ERROR-EXCP-WORK
        MOVE FIELD-NOT-NUMERIC-ERROR-MSG TO
              TYPE-OF-ERROR-EXCP-WORK
        PERFORM U000-WRITE-EXCEPTION-REPORT
        MOVE 'NO ' TO ARE-TRAN-RECORDS-VALID.
****************************************************************
*                                                              *
*  THE FUNCTION OF THIS MODULE IS TO EDIT A TYPE 2 TRANSACTION.*
*  IT IS ENTERED FROM AND EXITS TO THE B000-OBTAIN-VALID-LOAD- *
*  DATA MODULE.                                                *
*                                                              *
****************************************************************

 C020-EDIT-TYPE-2-RECORD.

     MOVE CUSTOMER-NUMBER-TRANSORT2 TO CUSTOMER-NUMBER-WORK.
     PERFORM U020-EDIT-CUSTOMER-NUMBER.
     IF CUSTOMER-DISCOUNT-TRANSORT2 IS NOT NUMERIC
        MOVE CUSTOMER-NUMBER-TRANSORT2 TO
              CUSTOMER-NUMBER-EXCP-WORK
        MOVE CUSTOMER-DISCOUNT-FIELD-MSG TO
              FIELD-IN-ERROR-EXCP-WORK
        MOVE FIELD-NOT-NUMERIC-ERROR-MSG TO
              TYPE-OF-ERROR-EXCP-WORK
        PERFORM U000-WRITE-EXCEPTION-REPORT
        MOVE 'NO ' TO ARE-TRAN-RECORDS-VALID
     ELSE
        IF CUSTOMER-DISCOUNT-TRANSORT2 IS GREATER THAN
              MAXIMUM-DISCOUNT-PERCENT
           MOVE CUSTOMER-NUMBER-TRANSORT2 TO
                 CUSTOMER-NUMBER-EXCP-WORK
           MOVE CUSTOMER-DISCOUNT-FIELD-MSG TO
                 FIELD-IN-ERROR-EXCP-WORK
           MOVE BAD-DISCOUNT-ERROR-MSG TO
                 TYPE-OF-ERROR-EXCP-WORK
           PERFORM U000-WRITE-EXCEPTION-REPORT
           MOVE 'NO ' TO ARE-TRAN-RECORDS-VALID.
     IF LAST-ACTIVITY-DATE-TRANSORT2 IS NOT NUMERIC
        MOVE CUSTOMER-NUMBER-TRANSORT2 TO
              CUSTOMER-NUMBER-EXCP-WORK
        MOVE LAST-ACTIVITY-DATE-FIELD-MSG TO
              FIELD-IN-ERROR-EXCP-WORK
        MOVE FIELD-NOT-NUMERIC-ERROR-MSG TO
              TYPE-OF-ERROR-EXCP-WORK
        PERFORM U000-WRITE-EXCEPTION-REPORT
        MOVE 'NO ' TO ARE-TRAN-RECORDS-VALID
     ELSE
        IF NOT VALID-MONTH
           MOVE CUSTOMER-NUMBER-TRANSORT2 TO
                 CUSTOMER-NUMBER-EXCP-WORK
           MOVE LAST-ACTIVITY-DATE-FIELD-MSG TO
                 FIELD-IN-ERROR-EXCP-WORK
           MOVE BAD-MONTH-ERROR-MSG TO TYPE-OF-ERROR-EXCP-WORK
           PERFORM U000-WRITE-EXCEPTION-REPORT
           MOVE 'NO ' TO ARE-TRAN-RECORDS-VALID
        ELSE
           SET MONTH-DAY-INDEX TO MONTH-LAST-ACTIVITY-TRANSORT2
           IF DAY-LAST-ACTIVITY-TRANSORT2 IS LESS THAN
                 FIRST-DAY-OF-MONTH-NUMBER OR
                 DAY-LAST-ACTIVITY-TRANSORT2 IS GREATER THAN
                 NUMBER-OF-DAYS-IN-MONTH (MONTH-DAY-INDEX)
              MOVE CUSTOMER-NUMBER-TRANSORT2 TO
                    CUSTOMER-NUMBER-EXCP-WORK
              MOVE LAST-ACTIVITY-DATE-FIELD-MSG TO
                    FIELD-IN-ERROR-EXCP-WORK
              MOVE BAD-DAY-ERROR-MSG TO TYPE-OF-ERROR-EXCP-WORK
              PERFORM U000-WRITE-EXCEPTION-REPORT
              MOVE 'NO ' TO ARE-TRAN-RECORDS-VALID.
****************************************************************
*                                                              *
*  THIS MODULE IS ENTERED TO VERIFY A VALID FIRST RECORD IN    *
*  THE PAIR OF RECORDS. IT IS ENTERED FROM AND EXITS TO THE    *
*  C000-OBTAIN-PAIR-OF-RECORDS MODULE.                         *
*                                                              *
****************************************************************

 D000-VERIFY-VALID-FIRST-REC.

     IF TRANSACTION-CODE-TRANS1-WORK = TRANSACTION-TYPE-ONE
        MOVE 'YES' TO IS-THE-FIRST-RECORD-VALID
     ELSE
        MOVE 'NO ' TO IS-THE-FIRST-RECORD-VALID
        MOVE CUSTOMER-NUMBER-TRANS1-WORK TO
              CUSTOMER-NUMBER-EXCP-WORK
        MOVE CUSTOMER-RECORDS-FIELD-MSG TO
              FIELD-IN-ERROR-EXCP-WORK
        IF TRANSACTION-CODE-TRANS1-WORK = TRANSACTION-TYPE-TWO
           MOVE TYPE2-BUT-NO-TYPE1-ERROR-MSG TO
                 TYPE-OF-ERROR-EXCP-WORK
           PERFORM U000-WRITE-EXCEPTION-REPORT
        ELSE
           MOVE BAD-TRANS-CODE-ERROR-MSG TO
                 TYPE-OF-ERROR-EXCP-WORK
           PERFORM U000-WRITE-EXCEPTION-REPORT.
****************************************************************
*                                                              *
*  THIS MODULE IS ENTERED TO VERIFY A VALID SECOND RECORD. IT  *
*  IS ENTERED FROM AND EXITS TO THE C000-OBTAIN-PAIR-OF-RECORDS*
*  MODULE.                                                     *
*                                                              *
****************************************************************

 D010-VERIFY-VALID-SECOND-REC.

     IF TRANSACTION-CODE-TRANS2-WORK = TRANSACTION-TYPE-TWO
        IF CUSTOMER-NUMBER-TRANS2-WORK =
              CUSTOMER-NUMBER-TRANS1-WORK
           MOVE 'YES' TO IS-THE-SECOND-RECORD-VALID
        ELSE
           MOVE CUSTOMER-NUMBER-TRANS1-WORK TO
                 CUSTOMER-NUMBER-EXCP-WORK
           MOVE CUSTOMER-RECORDS-FIELD-MSG TO
                 FIELD-IN-ERROR-EXCP-WORK
           MOVE TYPE1-BUT-NO-TYPE2-ERROR-MSG TO
                 TYPE-OF-ERROR-EXCP-WORK
           PERFORM U000-WRITE-EXCEPTION-REPORT
           MOVE CUSTOMER-NUMBER-TRANS2-WORK TO
                 CUSTOMER-NUMBER-EXCP-WORK
           MOVE CUSTOMER-RECORDS-FIELD-MSG TO
                 FIELD-IN-ERROR-EXCP-WORK
           MOVE TYPE2-BUT-NO-TYPE1-ERROR-MSG TO
                 TYPE-OF-ERROR-EXCP-WORK
           PERFORM U000-WRITE-EXCEPTION-REPORT
           MOVE 'NO ' TO ARE-THE-PAIR-OF-RECORDS-VALID
     ELSE
        MOVE CUSTOMER-NUMBER-TRANS1-WORK TO
              CUSTOMER-NUMBER-EXCP-WORK
        MOVE CUSTOMER-RECORDS-FIELD-MSG TO
              FIELD-IN-ERROR-EXCP-WORK
        MOVE TYPE1-BUT-NO-TYPE2-ERROR-MSG TO
              TYPE-OF-ERROR-EXCP-WORK
        PERFORM U000-WRITE-EXCEPTION-REPORT
        MOVE 'NO ' TO ARE-THE-PAIR-OF-RECORDS-VALID
        IF TRANSACTION-CODE-TRANS2-WORK NOT = TRANSACTION-TYPE-ONE
           MOVE CUSTOMER-NUMBER-TRANS2-WORK TO
                 CUSTOMER-NUMBER-EXCP-WORK
           MOVE CUSTOMER-RECORDS-FIELD-MSG TO
                 FIELD-IN-ERROR-EXCP-WORK
           MOVE BAD-TRANS-CODE-ERROR-MSG TO
                 TYPE-OF-ERROR-EXCP-WORK
           PERFORM U000-WRITE-EXCEPTION-REPORT.
****************************************************************
*                                                              *
*  THIS MODULE WRITES THE EXCEPTION REPORT. IT IS ENTERED FROM *
*  AND EXITS TO SEVERAL MODULES WITHIN THE PROGRAM (SEE        *
*  HIERARCHY CHART).                                           *
*                                                              *
****************************************************************

 U000-WRITE-EXCEPTION-REPORT.

     IF FIRST-PAGE-EXCP-RPT OR
           LINES-PRINTED-EXCP-RPT IS GREATER THAN
           PAGE-SIZE-EXCP-RPT
        PERFORM U010-PRINT-EXCP-RPT-HEADINGS
        MOVE SPACES TO EXCEPTION-REPORT-LINE
        MOVE CUSTOMER-NUMBER-EXCP-WORK TO CUSTOMER-NUMBER-EXCPRPT.
     IF CUSTOMER-NUMBER-EXCP-WORK NOT = PREVIOUS-CUSTOMER-NUMBER
        MOVE CUSTOMER-NUMBER-EXCP-WORK TO CUSTOMER-NUMBER-EXCPRPT
```

BEST COPY AVAILABLE

Source Listing (Part 5 of 5)

```
042050          MOVE CUSTOMER-NUMBER-EXCP-WORK TO PREVIOUS-CUSTOMER-NUMBER
042060          MOVE SPACE-TWO-LINES-EXCP-RPT TO PROPER-SPACING-EXCP-RPT.
042070      MOVE FIELD-IN-ERROR-EXCP-WORK TO FIELD-IN-ERROR-EXCPRPT.
042080      MOVE TYPE-OF-ERROR-EXCP-WORK TO TYPE-OF-ERROR-EXCPRPT.
042090      WRITE EXCEPTION-REPORT-LINE
042100              AFTER PROPER-SPACING-EXCP-RPT.
042110      ADD PROPER-SPACING-EXCP-RPT TO LINES-PRINTED-EXCP-RPT.
042120      MOVE SPACE-ONE-LINE-EXCP-RPT TO PROPER-SPACING-EXCP-RPT.
042130      MOVE SPACES TO EXCEPTION-REPORT-LINE.
042150*****************************************************************
042160*                                                               *
042170*  THIS MODULE PRINTS THE HEADINGS ON THE EXCEPTION REPORT.     *
042180*  IT IS ENTERED FROM AND EXITS TO THE U000-WRITE-EXCEPTION-    *
042190*  REPORT MODULE.                                               *
042200*                                                               *
043010*****************************************************************
043020
043030 W010-PRINT-EXCP-RPT-HEADINGS.
043040
043050      MOVE CURRENT-DATE TO DATE-HDG1-EXCP-RPT.
043060      MOVE PAGE-NUMBER-EXCP-RPT TO PAGE-NUMBER-HDG1-EXCP-RPT.
043070      WRITE EXCEPTION-REPORT-LINE FROM FIRST-HEADING-LINE-EXCP-RPT
043080              AFTER ADVANCING TO-THE-TOP-OF-THE-PAGE.
043090      ADD 1 TO PAGE-NUMBER-EXCP-RPT.
043100      WRITE EXCEPTION-REPORT-LINE FROM
043110              SECOND-HEADING-LINE-EXCP-RPT
043120              AFTER ADVANCING 2 LINES.
043130      WRITE EXCEPTION-REPORT-LINE FROM THIRD-HEADING-LINE-EXCP-RPT
043140              AFTER ADVANCING 1 LINES.
043150      MOVE SPACE-TWO-LINES-EXCP-RPT TO PROPER-SPACING-EXCP-RPT.
043160      MOVE ZEROS TO LINES-PRINTED-EXCP-RPT.
043180*****************************************************************
043190*                                                               *
043200*  THIS MODULE EDITS THE CUSTOMER NUMBER IN THE TRANSACTION     *
043210*  RECORDS. IT IS ENTERED FROM AND EXITS TO SEVERAL MODULES IN *
043220*  THE PROGRAM (SEE HIERARCHY CHART).                           *
043230*                                                               *
044010*****************************************************************
044020
044030 W020-EDIT-CUSTOMER-NUMBER.
044040
044050      IF (FIRST-FOUR-CHAR-CUST-NUMB-WORK IS NOT NUMERIC) OR
044060              (FIFTH-CHAR-CUST-NUMB-WORK IS NOT NUMERIC AND
044070               NOT VALID-CHECK-DIGIT-FOR-TEN)
044080          MOVE CUSTOMER-NUMBER-WORK TO CUSTOMER-NUMBER-EXCP-WORK
044090          MOVE CUSTOMER-NUMBER-FIELD-MSG TO
044100                  FIELD-IN-ERROR-EXCP-WORK
044110          MOVE FIELD-NOT-NUMERIC-ERROR-MSG TO
044120                  TYPE-OF-ERROR-EXCP-WORK
044130          PERFORM U000-WRITE-EXCEPTION-REPORT
044140          MOVE 'NO ' TO ARE-TRAN-RECORDS-VALID
044150      ELSE
044160          MOVE CUSTOMER-NUMBER-WORK TO
044170                  FIELD-TO-BE-CHECKED-MOD11-WORK
044180          CALL 'MOD11CHK' USING MOD-11-PASS-AREA-WORK
044190          IF CUST-NUMBER-DID-NOT-PASS-MOD11
044200              MOVE CUSTOMER-NUMBER-WORK TO CUSTOMER-NUMBER-EXCP-WORK
044210              MOVE CUSTOMER-NUMBER-FIELD-MSG TO
044220                      FIELD-IN-ERROR-EXCP-WORK
045010              MOVE FAILS-MOD11-ERROR-MSG TO
045020                      TYPE-OF-ERROR-EXCP-WORK
045030              PERFORM U000-WRITE-EXCEPTION-REPORT
045040              MOVE 'NO ' TO ARE-TRAN-RECORDS-VALID.
045060*****************************************************************
045070*                                                               *
045080*  THIS PARAGRAPH CONTAINS THE EXIT STATEMENT WHICH MARKS THE   *
045090*  END OF THE SORT OUTPUT PROCEDURE. IT IS BRANCHED TO BY A     *
045100*  GO TO STATEMENT IN THE A000-LOAD-CUSTOMER-MASTER-FILE        *
045110*  MODULE.                                                      *
045120*                                                               *
045130*****************************************************************
045140
045150 V001-SORT-PROCEDURE-EXIT.
045160
045170      EXIT.
```

```
045190 REST-OF-PROGRAM SECTION.
045200
046010*****************************************************************
046020*                                                               *
046030*  THIS MODULE IS ENTERED TO LIST THE CUSTOMER MASTER FILE. IT  *
046040*  IS ENTERED FROM AND EXITS TO THE A000-LOAD-CUSTOMER-MASTER-  *
046050*  FILE MODULE.                                                 *
046060*                                                               *
046070*****************************************************************
046080
046090 X000-LIST-CUSTOMER-MASTER.
046100
046110      READ CUSTOMER-MASTER-INPUT-FILE
046120              AT END
046130                  MOVE 'NO ' TO ARE-THERE-MORE-MSTR-RECORDS.
046140      PERFORM X001-WRITE-AND-READ
046150              UNTIL THERE-ARE-NO-MORE-MSTR-RECORDS.
046160
046170
046180
046190 X001-WRITE-AND-READ.
046200
047010      PERFORM Y000-WRITE-LIST-REPORT.
047020      READ CUSTOMER-MASTER-INPUT-FILE
047030              AT END
047040                  MOVE 'NO ' TO ARE-THERE-MORE-MSTR-RECORDS.
047060*****************************************************************
047070*                                                               *
047080*  THIS MODULE IS ENTERED TO WRITE THE MASTER LIST REPORT. IT   *
047090*  IS ENTERED FROM AND EXITS TO THE X000-LIST-CUSTOMER-MASTER   *
047100*  MODULE.                                                      *
047110*                                                               *
047120*****************************************************************
047130
047140 Y000-WRITE-LIST-REPORT.
047150
047160      IF FIRST-PAGE-MSTR-LIST OR
047170              LINES-PRINTED-MSTR-LIST IS GREATER THAN
047180              PAGE-SIZE-MSTR-LIST
047190          PERFORM Z000-PRINT-LIST-HEADINGS.
047200      MOVE SPACES TO MASTER-FILE-LIST-LINE.
048010      MOVE CUSTOMER-NUMBER-MSTRIN TO CUSTOMER-NUMBER-MSTRLIST.
048020      MOVE CUSTOMER-NAME-MSTRIN TO CUSTOMER-NAME-MSTRLIST.
048030      MOVE CUSTOMER-ADDRESS-MSTRIN TO CUSTOMER-ADDRESS-MSTRLIST.
048040      MOVE CUSTOMER-CITY-MSTRIN TO CUSTOMER-CITY-MSTRLIST.
048050      MOVE CUSTOMER-STATE-MSTRIN TO CUSTOMER-STATE-MSTRLIST.
048060      MOVE CUSTOMER-ZIP-CODE-MSTRIN TO CUSTOMER-ZIP-CODE-MSTRLIST.
048070      MOVE CUSTOMER-DISCOUNT-MSTRIN TO CUSTOMER-DISCOUNT-MSTRLIST.
048080      MOVE '%' TO PERCENT-SIGN-MSTRLIST.
048090      MOVE LAST-ACTIVITY-DATE-MSTRIN TO
048100              LAST-ACTIVITY-DATE-MSTRLIST.
048110      WRITE MASTER-FILE-LIST-LINE
048120              AFTER PROPER-SPACING-MSTR-LIST.
048130      ADD PROPER-SPACING-MSTR-LIST TO LINES-PRINTED-MSTR-LIST.
048140      MOVE SPACE-ONE-LINE-MSTR-LIST TO PROPER-SPACING-MSTR-LIST.
048160*****************************************************************
048170*                                                               *
048180*  THIS MODULE PRINTS THE HEADINGS ON THE MASTER LIST REPORT.   *
048190*  IT IS ENTERED FROM AND EXITS TO THE Y000-WRITE-LIST-REPORT   *
048200*  MODULE.                                                      *
049010*                                                               *
049020*****************************************************************
049030
049040 Z000-PRINT-LIST-HEADINGS.
049050
049060      MOVE CURRENT-DATE TO DATE-HDG1-MSTR-LIST.
049070      MOVE PAGE-NUMBER-MSTR-LIST TO PAGE-NUMBER-HDG1-MSTR-LIST.
049080      WRITE MASTER-FILE-LIST-LINE FROM
049090              FIRST-HEADING-LINE-MSTR-LIST
049100              AFTER ADVANCING TO-THE-TOP-OF-THE-PAGE.
049110      ADD 1 TO PAGE-NUMBER-MSTR-LIST.
049120      WRITE MASTER-FILE-LIST-LINE FROM
049130              SECOND-HEADING-LINE-MSTR-LIST
049140              AFTER ADVANCING 2 LINES.
049150      WRITE MASTER-FILE-LIST-LINE FROM
049160              THIRD-HEADING-LINE-MSTR-LIST
049170              AFTER ADVANCING 1 LINES.
049180      MOVE SPACE-TWO-LINES-MSTR-LIST TO PROPER-SPACING-MSTR-LIST.
049190      MOVE ZEROS TO LINES-PRINTED-MSTR-LIST.
```

BEST COPY AVAILABLE

TM 1

Format of Customer Master File

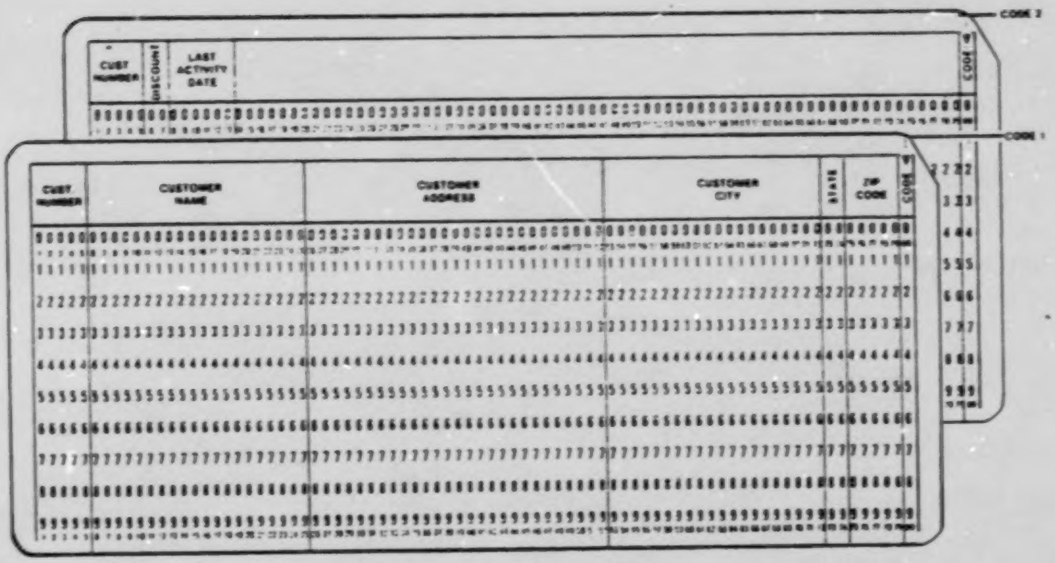
FIELD	POSITION	NUMBER OF DIGITS	ATTRIBUTES
Not used	1	1	Blank
Customer Number	2 - 6	5	Alphanumeric
Customer Name	7 - 26	20	Alphanumeric
Customer Address	27 - 53	27	Alphanumeric
Customer City	54-73	20	Alphanumeric
Customer State	74 - 75	2	Alphanumeric
Customer Zip Code	76 - 80	5	Numeric
Customer Discount	81 - 82	2	Numeric
Last Activity Date	83 - 88	6	Numeric

File Information:
Record Length – 88 characters
Blocking Factor – 20 records per block
Storage Medium – Disk
Organization – Indexed Sequential
Access – Sequential

TM 2

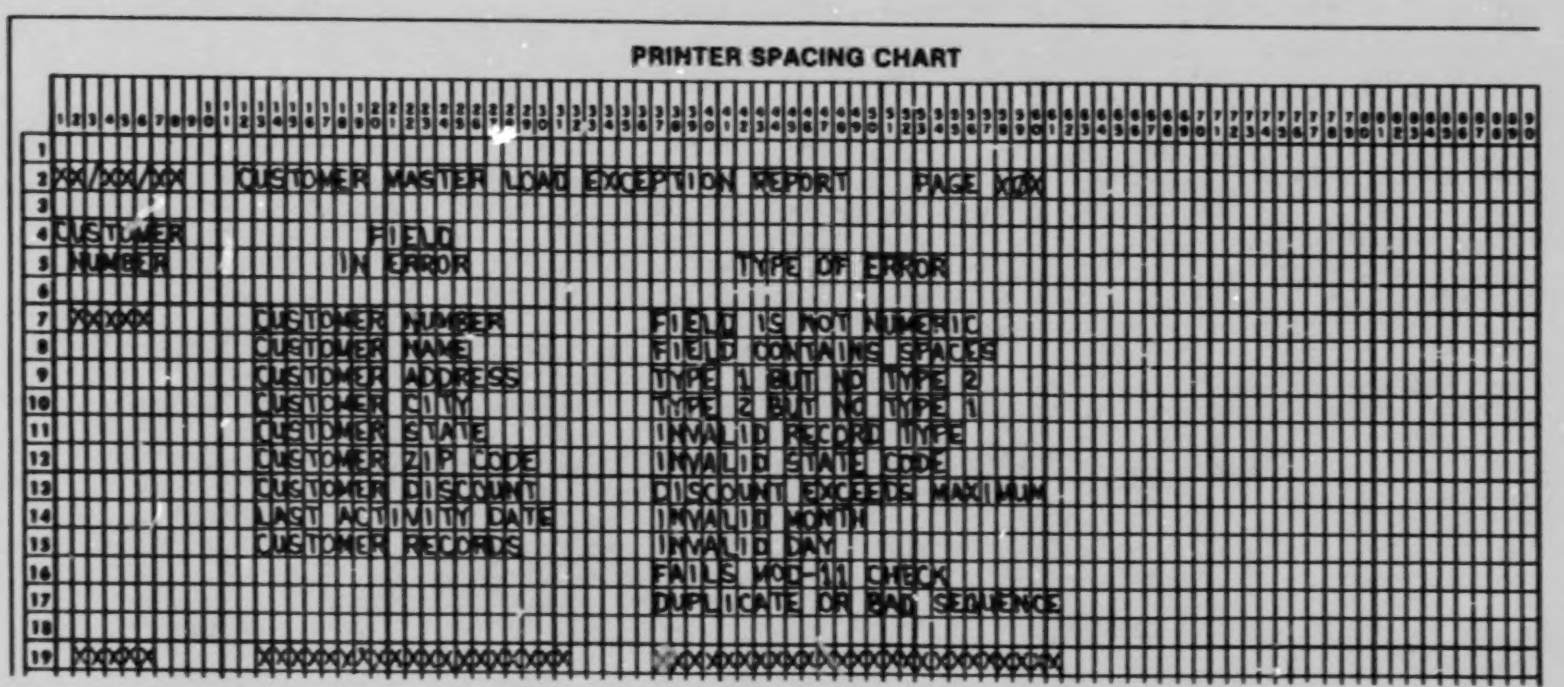
BEST COPY AVAILABLE

Format of Input Records


CODE 2
CUST NUMBER
DISCOUNT
LAST ACTIVITY DATE
CODE
CODE 1
CUST NUMBER
CUSTOMER NAME
CUSTOMER ADDRESS
CUSTOMER CITY
STATE
ZIP CODE
CODE

TM 3

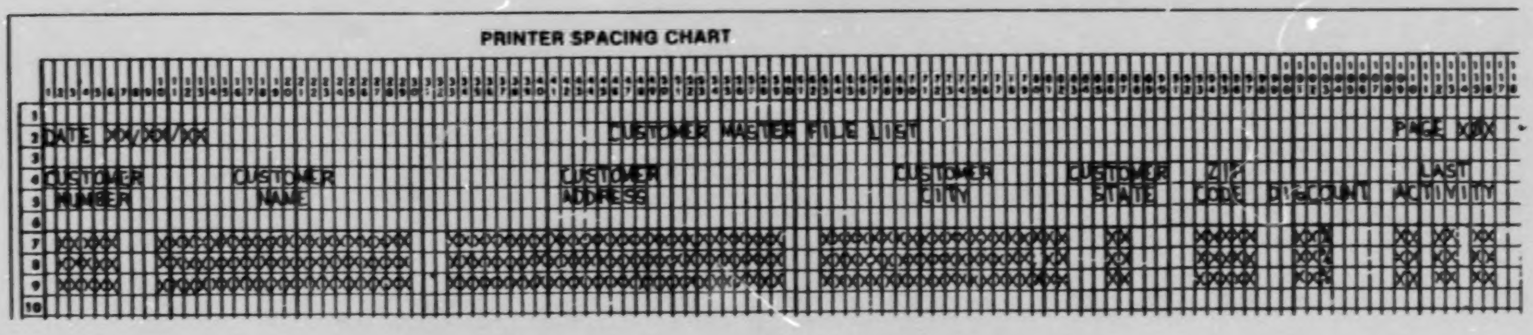
Format of the Exception Report



PRINTER SPACING CHART

```
XX/XX/XX     CUSTOMER MASTER LOAD EXCEPTION REPORT     PAGE XXX

CUSTOMER                FIELD
 NUMBER               IN ERROR                    TYPE OF ERROR

 XXXXX        CUSTOMER NUMBER               FIELD IS NOT NUMERIC
              CUSTOMER NAME                 FIELD CONTAINS SPACES
              CUSTOMER ADDRESS              TYPE 1 BUT NO TYPE 2
              CUSTOMER CITY                 TYPE 2 BUT NO TYPE 1
              CUSTOMER STATE                INVALID RECORD TYPE
              CUSTOMER ZIP CODE             INVALID STATE CODE
              CUSTOMER DISCOUNT             DISCOUNT EXCEEDS MAXIMUM
              LAST ACTIVITY DATE            INVALID MONTH
              CUSTOMER RECORDS              INVALID DAY
                                            FAILS MOD-11 CHECK
                                            DUPLICATE OR BAD SEQUENCE

 XXXXX        XXXXXXXXXXXXXXXXXXXXXXX       XXXXXXXXXXXXXXXXXXXXXXXXX
```

Format of the Customer Master File Listing

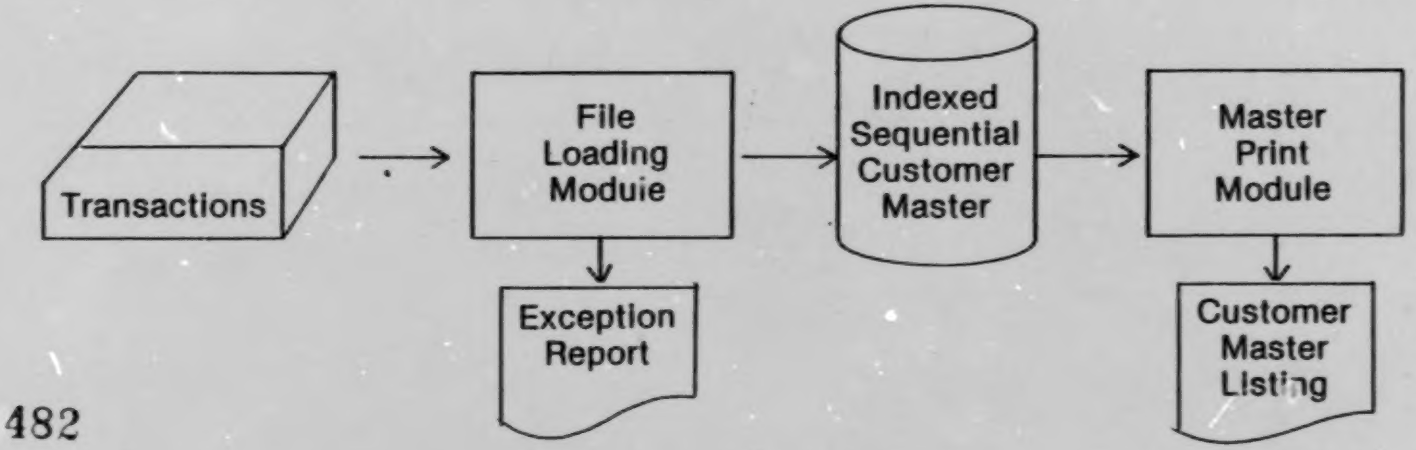


PRINTER SPACING CHART

```
DATE XX/XX/XX                                 CUSTOMER MASTER FILE LIST                                    PAGE XXX

CUSTOMER     CUSTOMER                  CUSTOMER                        CUSTOMER      CUSTOMER  ZIP                  LAST
 NUMBER        NAME                    ADDRESS                           CITY          STATE   CODE  DISCOUNT     ACTIVITY

 XXXXX   XXXXXXXXXXXXXXXXXXXX   XXXXXXXXXXXXXXXXXXXXXXXXX   XXXXXXXXXXXXXXXXXXXX   XX   XXXXX   X.XX   XX XX XX
 XXXXX   XXXXXXXXXXXXXXXXXXXX   XXXXXXXXXXXXXXXXXXXXXXXXX   XXXXXXXXXXXXXXXXXXXX   XX   XXXXX   X.XX   XX XX XX
 XXXXX   XXXXXXXXXXXXXXXXXXXX   XXXXXXXXXXXXXXXXXXXXXXXXX   XXXXXXXXXXXXXXXXXXXX   XX   XXXXX   X.XX   XX XX XX
```

TM 5

Overview of Sample Program to Load Indexed Sequential File



Transactions

File Loading Module

Exception Report

Indexed Sequential Customer Master

Master Print Module

Customer Master Listing

IPO Chart – Load Customer Master File

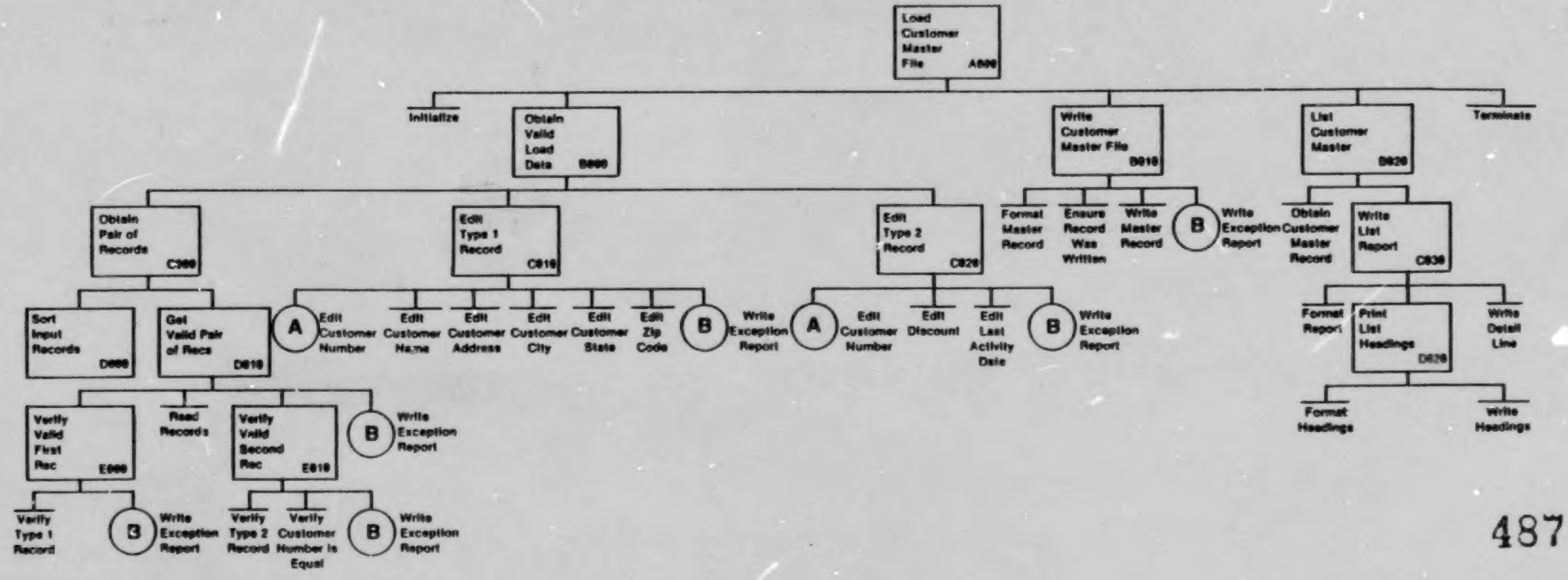
Input/Processing/Output Chart

DATE: May 2
PROGRAMMER: Jane Smith
PROGRAM: Load Customer Master FIle
MODULE FUNCTION: Load the Customer Master File
REF: A000
MODULE NAME: Load Customer Master File

INPUT	PROCESSING	REF:	OUTPUT
1. Customer	1. Initialize		1. Customer
Transaction File	2. Obtain valid data to load	B000	Master File
	master file	B010	2. Customer
	3. Write customer master file	B020	Master File List
	4. List customer master file		
	5. Terminate		

BEST COPY AVAILABLE

Hierarchy Chart for Sample Program


Load Customer Master File A000
Initialize
Obtain Valid Load Data B000
Write Customer Master File B010
List Customer Master B020
Terminate
Obtain Pair of Records C000
Edit Type 1 Record C010
Edit Type 2 Record C020
Format Master Record
Ensure Record Was Written
Write Master Record
B
Write Exception Report
Obtain Customer Master Record
Write List Report C030
Sort Input Records D000
Get Valid Pair of Recs D010
A
Edit Customer Number
Edit Customer Name
Edit Customer Address
Edit Customer City
Edit Customer State
Edit Zip Code
B
Write Exception Report
A
Edit Customer Number
Edit Discount
Edit Last Activity Date
B
Write Exception Report
Format Report
Print List Headings D020
Write Detail Line
Verify Valid First Rec E000
Read Records
Verify Valid Second Rec E010
B
Write Exception Report
Format Headings
Write Headings
Verify Type 1 Record
B
Write Exception Report
Verify Type 2 Record
Verify Customer Number Is Equal
B
Write Exception Report

Select Sentence – Indexed Sequential File

```
003060     SELECT CUSTOMER-MASTER-OUTPUT-FILE
003070           ASSIGN TO DA-I-MSTROUT
003080           ACCESS IS SEQUENTIAL
003090           RECORD KEY IS CUSTOMER-NUMBER-MSTROUT.
```

Environment and Data Division Entries for Sequential Retrieval

Environment Division

```
003120         SELECT CUSTOMER-MASTER-INPUT-FILE
003130                ASSIGN TO DA-I-MSTROUT
003140                ACCESS IS SEQUENTIAL
003150                RECORD KEY IS CUSTOMER-NUMBER-MSTRIN.
```

Data Division

```
007040  FD     CUSTOMER-MASTER-INPUT-FILE
007050             BLOCK CONTAINS 20 RECORDS
007060             RECORD CONTAINS 88 CHARACTERS
007070             LABEL RECORDS ARE STANDARD
007080             DATA RECORD IS CUSTOMER-MASTER-INPUT-RECORD.
007090
007100  01     CUSTOMER-MASTER-INPUT-RECORD.
007110         05     FILLER                             PIC X.
007120         05     CUSTOMER-NUMBER-MSTRIN             PIC X(5).
007130         05     CUSTOMER-NAME-MSTRIN               PIC X(20).
007140         05     CUSTOMER-ADDRESS-MSTRIN            PIC X(27).
007150         05     CUSTOMER-CITY-MSTRIN               PIC X(20).
007160         05     CUSTOMER-STATE-MSTRIN              PIC XX.
007170         05     CUSTOMER-ZIP-CODE-MSTRIN           PIC X(5).
007180         05     CUSTOMER-DISCOUNT-MSTRIN           PIC XX.
007190         05     LAST-ACTIVITY-DATE-MSTRIN          PIC X(6).
```

File Definition of Customer Master File

```
004100 FD  CUSTOMER-MASTER-OUTPUT-FILE
004110       BLOCK CONTAINS 20 RECORDS
004120       RECORD CONTAINS 88 CHARACTERS
004130       LABEL RECORDS ARE STANDARD
004140       DATA RECORD IS CUSTOMER-MASTER-OUTPUT-RECORD.
004150
004160 01  CUSTOMER-MASTER-OUTPUT-RECORD.
004170     05  FILLER                      PIC X.
004180     05  CUSTOMER-NUMBER-MSTROUT     PIC X(5).
004190     05  CUSTOMER-NAME-MSTROUT       PIC X(20).
004200     05  CUSTOMER-ADDRESS-MSTROUT    PIC X(27).
005010     05  CUSTOMER-CITY-MSTROUT       PIC X(20).
005020     05  CUSTOMER-STATE-MSTROUT      PIC XX.
005030     05  CUSTOMER-ZIP-CODE-MSTROUT   PIC X(5).
005040     05  CUSTOMER-DISKOUNT-MSTROUT   PIC XX.
005050     05  LAST-ACTIVITY-DATE-MSTROUT  PIC X(6).
```

INDEXED SEQUENTIAL FILES I

UNIT XII

ASSIGNMENT SHEET I: Understanding the Sample Program Specifications

1. Explain the purpose of this program.

2. Why is the Customer Master Output File outputted sequentially?

3. Explain the purpose for each file named in the File-Control paragraph of the program listing.

INDEXED SEQUENTIAL FILES I

UNIT XII

ANSWERS TO ASSIGNMENT SHEETS

Assignment Sheet I

1. This program creates the master file for use in the next unit as an indexed sequential master file.

2. The file is outputted sequentially to simplify the program. (The JCL actually sets up the indexed access pathways.)

3.
    a. Customer-Trans-Input-File - used to read input transactions
    b. Customer-Master-Output-File - used to hold the newly created master data for the next unit's program
    c. Customer-Master-Input-File - used to print the master output file
    d. Transaction-Sort-Work-File - used by the sort for utility sorting

INDEXED SEQUENTIAL FILES I

UNIT XII

EXAMINATION

1. Match the definitions in Column A with the corresponding phrase in Column B.

	Column A		Column B
____ a.	Going directly to a record for access	1.	Indexed sequential
____ b.	Allows both random and sequential access	2.	Random access
____ c.	Software package	3.	Indexed sequential access method (ISAM)
____ d.	Pairs of keys and addresses	4.	Index

2. Explain the term "indexed sequential file" and its advantages.

3. List two major functions from the Load Customer Master File Module's IPO chart.

True and False:

T F 1. The transaction input file for this program is in random order.

T F 2. In the pseudocode, the Get Valid Pair of Recs Modules checks that there is both a type "1" record and a type "2" record following it.

T F 3. In the listing of the program, the ACCESS is SEQUENTIAL clause is used to describe the access mode of the master file.

INDEXED SEQUENTIAL FILES I

UNIT XII

ANSWERS TO EXAMINATION

1.
   a. 2
   b. 1
   c. 3
   d. 4

2. Indexed sequential file refers to a file which can be accessed both sequentially and randomly through an index. The advantage is that voluminous data without need for rapid response time and less voluminous data with rapid response time requirements can both be processed by this type of file.

3. Any two:
   a. Initialize files.
   b. Obtain valid data.
   c. Write master file data.
   d. List customer master file data.
   e. Terminate.

True and False

a. F
b. T
c. T

INDEXED SEQUENTIAL FILES II

UNIT XIII

UNIT OBJECTIVES

After studying this unit, students should be able to apply elementary indexed sequential file access concepts to updating of indexed sequential files in writing COBOL programs.

SPECIFIC OBJECTIVES

After studying this unit, students should be able to:

1. Match terms and objectives.
2. Explain general considerations in random updates of indexed sequential fields.
3. Recognize correct statements from the sample program's specifications.
4. Write COBOL syntactical constructs used in random indexed sequential file updates.
5. Explain statements from the sample program's design.
6. Apply preceding objectives in writing indexed sequential file update programs.

INDEXED SEQUENTIAL FILES II

UNIT XIII

SUGGESTED ACTIVITIES

I. Instructor

   A. Make transparencies; prepare lecture using this curriculum guide and supplemental materials.

   B. Provide students with objectives; discuss objectives.

   C. Make parallel assignments for reading and programming from the text(s).

   D. Provide students with information and assignment sheets; discuss these sheets.

   E. Provide students with additional supplementary information and exercises.

   F. Provide students with job sheets and make assignments.

   G. Give examination.

II. Student

   A. Read objectives.

   B. Study information sheets and text assignments.

   C. Complete assignment sheets and suplementary exercises.

   D. Complete assigned job sheet(s).

   E. Complete examination.

INSTRUCTIONAL MATERIALS

I. Unit Content

   A. Objectives

   B. Notes to the instructor

   C. Information sheets

D. Handouts

   1. HO 1: Complete Hierarchy Chart
   2. HO 2: Pseudocode Specifications (Part 1 through 12)
   3. HO 3: Source Listing (Part 1 through 7)

E. Transparency Masters

   1. TM 1: Overview of Random Update
   2. TM 2: Format of Customer Master File
   3. TM 3: Format of Customer Transaction Input File
   4. TM 4: Format of Transaction Report

F. Assignment Sheet

   1. AS 1: Understanding the Sample Program

G. Answers to assignment sheet

H. Job Sheet 1: Writing A Complete Indexed Sequential Program for Payroll

   1. Figure JS 1-1: Overview of Application to Create, Update, and Print Indexed Sequential Payroll Master Files
   2. Figure JS 1-2: Hierarchy Chart for Payroll Indexed Sequential File Update Program, H 2
   3. Figure JS 1-3: Pseudocode Indexed Sequential File Update H 2
   4. Figure JS 1-4: Prewritten Divisions for Indexed Sequential Payroll Application
   5. Figure JS 1-5: Output from Program H 2
   6. Figure JS 1-6: Output Showing Updating and Debugging Traces Parts 1 and 2)

I. Job Sheet 1: Appendix: Sample Application Indexed Sequential Update Program for Inventory

   1. Figure JS 1-7: VSAM Setup Program for an Inventory Application
   2. Figure JS 1-8: Listing of Sample Indexed Sequential File Update Program for Inventory

J. Answers to Job Sheet I:

    1. Program H 1 - Setup Solution for Payroll (Non-VSAM portion)

    2. Program H 2 - Indexed Sequential Solution for Payroll (Procedure Division)

    3. Program H 3 - (changed routines and printouts)

K. Examination

L. Answers to examination

INDEXED SEQUENTIAL FILES II

UNIT XIII

NOTES TO INSTRUCTOR

You may find it advantageous to start the job in this unit during coverage of unit XII.

The sample program in the text of the unit was written for 1965 COBOL; the job sheet sample was written for 1974 COBOL. The major difference is in the use of the DELETE statement. This statement will be explained in detail in the manufacturer's COBOL manual for the computer you are using.

In the job for this unit you will need either a utility program or your own program (written in JCL) to create the Indexed Sequential Master File index, to load the contents of the master file, and to delete any file changes made by previous runs of students. A sample utility program is given for a large scale IBM environment using the MVS operating system. Your computer center should be able to help you adapt this program to your own installation's requirements.

Students may test their programs from cards or terminals, so it is particularly important to provide a short set of test transactions which will be appropriate for either media.

INDEXED SEQUENTIAL FILES II

UNIT XIII

INFORMATION SHEETS

I. Terms and definitions

A. **OLRT** - "on line real time" refers to fast access and processing on a computer in contrast to batch or delayed processing.

B. **Random update** - processing of master files in which records are updated in the order that transactions are presented for processing, usually randomly

II. Introduction to random access file updating

A. Overview:

1. TM 1 gives an overview of the random access file updating.

a. In the TM at point A, notice that the transactions, while depicted arriving on cards, usually come from remote, on-line interactive terminals.

b. There is no batching, sorting, or holding of transactions.

c. Transactions are processed one at a time as they occur, which is generally in random order.

d. The response to the transaction (processing, file access, master record update, etc.) must be fast. Fast here generally means at normal conversational speed. For technical reference, this is often called On-line Real Time (OLRT) processing.

e. In the TM at points B and D, also notice the two-headed arrow going to the master file and the absence of the "old" and "new" copies of the master file as was true in sequential updates. The reason for only one copy is that the master records are copied back in their old place instead of to a new file.

f. There is no automatic backup copy of the master file, instead records are retrevied, updated from the contents of the transaction and then rewritten back in place, thereby destroying the old copy of the affected master records.

g. In the TM at point C, a report is depicted - often this report is a log of transactions, which is written on disk and then printed at a later time in order that the file could be recreated in case of catastrophic failure. This report might also contain master record information.

B. Additional items

1. The type of updating to be done in this unit is random updating; but indexed sequential files allow both random and sequential updates, and therein lies their popularity.

2. Although indexed sequential files can be concurrently updated both sequentially and randomly, this unit will be simplified by only randomly updating the files.

3. The master file to be updated will also exist.

4. Transaction types will be additions, deletions, and changes. Another type called postings, which are generally financial transactions, are not considered in this unit.

III. Overview of sample problem: random file updating

A. Inputs and outputs

TM 2 gives the format of the master file which is the same in the preceding unit.

B. TM 3 gives the format of the transactions, which are assumed to be on cards for simplification. These transactions are also the same as in the preceding unit.

C. TM 4 gives the format of the transaction report.

D. The programming specifications for the processing in the sample program which is to random update an indexed sequential file are as follows:

1. The program should read transaction records from the Customer Transaction Input File and use these records to update the Customer Master File. The updating is to be done randomly.

2. The Customer Transaction Input Records must be sorted into an ascending sequence by Transaction Code within Customer Number.

3. Three types of processing can take place. A record can be added to the Customer Master File; a record can be deleted from the Customer Master File; and a record in the Customer Master File can be changed.

a. Additions - Records can be added to the Customer Master File provided there is not a record with the same Customer Number already in the file. When a record is added to the Customer Master File, the Customer Number, Customer Name, Customer Address, Customer City, Customer State, Customer Zip Code, Customer Discount, and Last Activity Date will be taken from a Type "1" Transaction Record and a Type "2" Transaction Record and placed in new Customer Master Record. Therefore, in order for a valid addition to take place, there must be both a Type 1 transaction and a Type 2 transaction with the same

Customer Number. If there is a Type 1 transaction without a Type 2 transaction with the same Customer Number, or if there is a Type 2 transaction without a corresponding Type 1 transaction record, then an addition cannot take place; instead, a message should be written on the Update Transaction Report indicating a Type 1 record without a Type 2, or a Type 2 record without a Type 1.

If there is a Type 1 record followed by a Type 2 record with the same Customer Number, then the data in those records can be used to add a new record to the Customer Master File provided the data within the records pass the editing specified later in these specifications.

b. Deletions - A record can be deleted from the Customer Master File if the Transaction Code in the transaction record is equal to the value "5" and there is a corresponding record in the Customer Master File; that is, a record with the same Customer Number as that found in the transaction record. In order to cause a record to be deleted, the COBOL figurative constant High-Values, which is hexadecimal "FF," should be moved to the first position of the record to be deleted. The record should then be rewritten on the Customer Master File. **Note some COBOL compilers do not use this convention; check with your instructor.**

c. Changes - Fields within a Customer Master Record contained in the Customer Master File can be changed if the Transaction Code in the Transaction Record is equal to "3" or "4." A Transaction Record with a code equal to "3" can be used to change the Customer Address, Customer City, Customer State, and Customer Zip Code. A Transaction Record with a code equal to "4" can be used to change the Customer Discount and the Last Activity Date. In order for a change to occur, the Transaction Code must be equal to "3" or "4" and a matching Customer Master Record must be found in the Customer Master File. If a matching Customer Master Record is not found in the Customer Master File when the Transaction Code is equal to "3" or "4," then an error has occurred and an entry should be made on the Update Transaction Report.

Code "3" Transaction:

1) If the Customer Address in the Transaction Record is not blank, then the Customer Address in the Customer Master Record should be replaced by the Customer Address in the Transaction Record.

2) If the Customer City in the Transaction Record is not blank, then the Customer City in the Customer Master Record should be replaced by the Customer City in the Transaction Record.

3) If the Customer State in the Transaction Record is not blank, then the Customer State in the Customer Master Record should be replaced by the Customer State in the Transaction Record.

4) If the Customer Zip Code in the Transaction Record is not blank, then the Customer Zip Code in the Customer Master Record should be replaced by the Customer Zip Code in the Transaction Record.

Code "4" Transaction:

1) If the Customer Discount in the Transaction Record is not blank, then the Customer Discount in the Customer Master Record should be replaced by the Customer Discount in the Transaction Record.

2) If the Last Activity Date in the Transaction Record is not blank, then the Last Activity Date in the Customer Master Record should be replaced by the Last Activity Date in the Transaction Record.

4. Before any Transaction Records can be used for additions, deletions, or changes, they must be edited to ensure that they contain valid data. The editing which must be performed is specified below.

Additions:

a. Customer Number - The Customer Number must be numeric.

b. Customer Address, Customer City, Customer State - The Customer Address, Customer City, and Customer State fields must not be blank.

c. Customer State - The State Code must be one of the valid 51 state codes (50 states and the District of Columbia).

d. Zip Code - The Zip Code must be numeric.

e. Customer Discount - The Customer Discount field must be numeric and must not be greater than 20%.

f. Last Activity Date - The Last Activity Date must be numeric, and the month must not be less than 01 or greater than 12. The day of the month in the Transaction Record must not exceed the maximum number of days in that month.

Deletions:

a. Customer Number - The Customer Number must be numeric.

Changes:

a. Customer Number - The Customer Number must be numeric.

b. Customer State - The State Code must be one of the valid 51 state codes (50 states and the District of Columbia).

c. Zip Code - The Zip Code must be numeric.

d. Customer Discount - The Customer Discount field must be numeric and must not be greater than 20%.

e. Last Activity Date - The Last Activity Date must be numeric, and the month must not be less than 01 or greater than 12. The day of the month in the Transaction Record must not exceed the maximum number of days in that month.

f. If a change is made to the State Code in the Master Record, then a change must also be made to the Zip Code. Thus, a Transaction Record with a State Code present but a Zip Code absent is in error.

5. For Addition Records, all fields must be checked and must be valid before the record is added to the Customer Master File. The Addition Records must have both a Type 1 record and a Type 2 record with the same Customer Number before the editing takes place.

6. For Change Records, a Type 3 record will be checked for Customer Number, Customer State, and Customer Zip Code provided the field in the record contains data. For example, it is valid to have a Type 3 record contain only the Customer Zip Code. In that event, only the Zip Code would be edited. The other fields would not be edited because there is no data to edit. Thus, if a field contains data which are going to be used to update the Customer Master Record, the data should be edited according to the Editing Rules set forth in #5 above. The Customer Number will always be checked.

   A Type 4 record will be checked for Customer Discount and Last Activity Date, again provided there are data in the field. For example, if only the Last Activity Date is to be changed by a Type 4 record, then only the Last Activity Date field should be edited because the Customer Discount field will be blank. The Customer Number must be edited for each Type 4 record.

   If any field within a Change Record is found to be in error, then the entire record is considered invalid, and none of the fields can be used to change the Master Record.

7. The Update Transaction Report contains a record of each Transaction Record which is processed, whether it causes a Change, Deletion, or Addition to take place or whether it contains an error. The fileds on the Update Transaction are explained below:

   a. Customer Number - The Customer Number will be printed for each Transaction Record which is processed. It should be group-indicated since more than one error or more than one type of transaction can occur for the same Customer.

b. Action Taken - This field indicates the type of updating which took place in the Master Record. The constant "CHANGE" should be printed when the Master Record is changed; the constant "ADDITION" should be printed when a record is added to the Customer Master File; the constant "DELETION" is printed when the Customer Master Record is deleted. The field should be left blank when a Transaction Records contains an error.

c. Field Changed or Field in Error - This field contains the name of the field which was changed by a Change Transaction or which contained an error when the Transaction Record was edited. It is left blank for Additions and Deletions. The following are the constants which should be used to identify the fields:

CUSTOMER NUMBER - Customer Number field
CUSTOMER NAME - Customer Name field
CUSTOMER ADDRESS - Customer Address field
CUSTOMER CITY - Customer City field
CUSTOMER STATE - Customer State field
CUSTOMER ZIP CODE - Customer Zip Code field
CUSTOMER DISCOUNT - Customer Discount field
LAST ACTIVITY DATE - Last Activity Date field
CUSTOMER RECORDS - Used when have Type 1 but No Type 2 Record, or Type 2 but No Type 1 Record when adding records
TRANSACTION CODE - Used when a change or delete record does not have a matching master record; when an addition of a currently existing Master Record is attempted; or when the Transaction Code is invalid (not 1-5)

d. Type of Error - This field is used to identify the type of error which was found in the editing processing. It is blank for Additions, Deletions, and Changes. The allowable error messages are listed below:

FIELD CONTAINS SPACES - This message is printed whenever a field which is supposed to contain data does not.
INVALID STATE CODE - This message is printed when the State Code in the Transaction Record does not contain one of the valid 51 codes.
FIELD IS NOT NUMERIC - This message is printed when a field which is supposed to contain numeric data does not.
DISCOUNT EXCEEDS MAXIMUM - This message is printed when the Customer Discount in a Transaction Record is greater than the maximum customer discount allowed.
INVALID MONTH - This message is printed when the month in the Last Activity Date is not 01-12.
INVALID DAY - This is printed when the day in the Last Activity Date is Less then 01 or greater than the number of days in the month specified in the Last Activity Date.
TYPE 1 BUT NO TYPE 2 - This message is printed when a Type 1 record is read but there is no Type 2 record with the same Customer Number.
TYPE 2 BUT NO TYPE 1 - This message is printed when a Type 2 record is read but there is not Type 1 record with the same Customer Number.

INVALID RECORD TYPE - This message is printed when the Transaction Code in the Transaction Record is not equal to 1, 2, 3, 4, or 5.
ADDITION OF DUPLICATE RECORD - This message is printed when an attempt is made to add a record with a Customer Number equal to a Customer Number already stored in the Customer Master File.
STATE CHANGED-ZIP NOT - This message is printed when a Type 3 change record contains a State Code, indicating that the State Code in the Customer Master Record is to be changed but does not contain data in the Zip Code field to indicate that the Zip Code is to be changed also.
NO MASTER RECORD - This message is printed whn a Change record or a Delete record contains a Customer Number for which there is not a record in the Customer Master File.

e. Old Master Record - This field will contain the contents of the field in the master record before a change is made to the Master Record. It will be blank for errors, additions, and deletions.

f. New Master Record - This field will contain the contents of a changed field after the change takes place. It will be blank for errors, additions, and deletions.

8. The Update Transaction Report should be formatted as illustrated in the accompanying printer spacing chart. The maximum number of detail lines for a single page should be 40 lines.

9. The program should be written in COBOL.

IV. Defining an indexed sequential file in COBOL

A. Environment Division

1. New clauses are:

   ACCESS IS RANDOM
   RECORD KEY IS data-name-1
   NOMINAL KEY IS data-name-2

2. The ACCESS IS RANDOM clause tells the compiler that the file is to be randomly accessed

3. The RECORD KEY IS clause identifies the field within the master record which is used as the key field.

4. The NOMINAL KEY IS clause is used to specify a field which is set to the contents of the master key to be retrieved; i.e., this field tells the ISAM which master record to retrieve. This field must be set by the programmer's code before each retrieval, normally the transaction's key is moved to this nominal key to set the nominal key to the value of the master key which is to be retrieved. Note that some COBOL compilers do not use the NOMINAL KEY clause but simply require

that the RECORD KEY is set before each retrieve, check with your instructor for information on your compiler.

B. Procedure Division

1. Open statement is new:

OPEN I-O file-name-I

which tells the compiler that there will be both input and output to the master file.

2. INVALID KEY clause with the READ statement is the same as in the preceding unit

3. REWRITE statement.

a. In random update, after a record is retrieved it should be rewritten back in place - the REWRITE accomplishes this rewrite.

b. The rewrite is controlled by the contents of the NOMINAL KEY (if that clause is used) or by the contents of the RECORD KEY IS field.

4. The write statement

a. Used only to add new records to the file

b. Use with INVALID KEY clause. This clause will be activated if the record key already exists on the file.

V. Sample program design

A. Hierarchy chart and IPO charts for the sample program

1. HO I gives the complete hierarchy chart for the sample program.

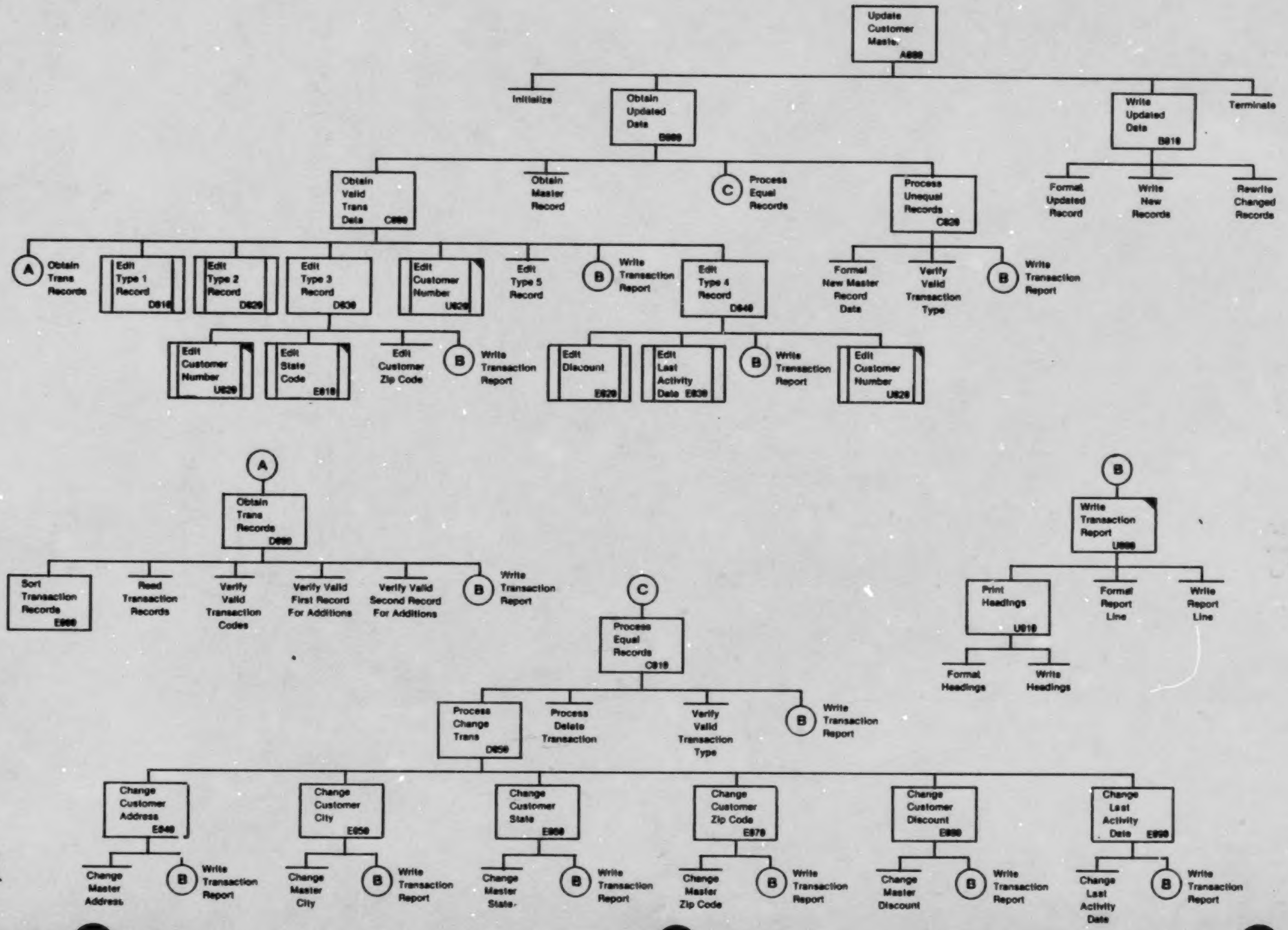
2. IPO charts, if used, will be provided by your instructor.

B. Pseudocode specifications for the sample program (HO 2)

C. Source listing (HO 3)

HO 1

Complete Hierarchy Chart


Update Customer Maste. A000
Initialize
Obtain Updated Data B000
Write Updated Data B010
Terminate
Obtain Valid Trans Data C000
Obtain Master Record
C
Process Equal Records
Process Unequal Records C020
Format Updated Record
Write New Records
Rewrite Changed Records
A
Obtain Trans Records
Edit Type 1 Record D010
Edit Type 2 Record D020
Edit Type 3 Record D030
Edit Customer Number U020
Edit Type 5 Record
B
Write Transaction Report
Edit Type 4 Record D040
Format New Master Record Data
Verify Valid Transaction Type
B
Write Transaction Report
Edit Customer Number U020
Edit State Code E010
Edit Customer Zip Code
B
Write Transaction Report
Edit Discount E020
Edit Last Activity Date E030
B
Write Transaction Report
Edit Customer Number U020
A
Obtain Trans Records D000
Sort Transaction Records E000
Read Transaction Records
Verify Valid Transaction Codes
Verify Valid First Record For Additions
Verify Valid Second Record For Additions
B
Write Transaction Report
C
Process Equal Records C010
Process Change Trans D050
Process Delete Transaction
Verify Valid Transaction Type
B
Write Transaction Report
B
Write Transaction Report U000
Print Headings U010
Format Report Line
Write Report Line
Format Headings
Write Headings
Change Customer Address E040
Change Customer City E050
Change Customer State E060
Change Customer Zip Code E070
Change Customer Discount E080
Change Last Activity Date E090
Change Master Address
B
Write Transaction Report
Change Master City
B
Write Transaction Report
Change Master State
B
Write Transaction Report
Change Master Zip Code
B
Write Transaction Report
Change Master Discount
B
Write Transaction Report
Change Last Activity Date
B
Write Transaction Report

HO 2

Pseudocode Specifications

(Part 1 of 12)

Pseudocode Specifications

DATE: June 17	PROGRAMMER: Jane Austin	PROGRAM: Update Customer Master File
MODULE FUNCTION: Update Customer Master File	REF: A000	MODULE NAME: Update Customer Master

```
PSEUDOCODE                                      REF:
Open the files
Obtain updated data                             B000
PERFORM UNTIL there are no more
        transaction records
    Write updated master file
    Obtain updated data                         B010
ENDPERFORM                                      B000
Close files
Stop run
```

FILES, RECORDS, FIELDS REQUIRED

Customer master file (I-O file)
Transaction report file
No more tran records indicator

Pseudocode Specifications (Part 2 of 12)

Pseudocode Specifications

DATE: June 17	PROGRAMMER: Jane Austin	PROGRAM: Update Customer Master File
MODULE FUNCTION: Obtain Updated Data	REF: B000	MODULE NAME: Obtain Updated Data

PSEUDOCODE — **REF:**

```
Set updated data obtained indicator to no
        data  obtained                          C000
Obtain valid transaction data
PERFORM UNTIL updated data has been
        obtained or there are no more
        transaction  records
  Set add indicator to neutral
  Set delete indicator to neutral
  Set change indicator to neutral
  Set master found indicator to "yes"
  Move customer number to master key area
  Read the master file
  IF a record was found
    Move master record customer number to
      master record being processed work
        area
  ELSE
    Move spaces to master record being          C020
      processed work area
  ENDIF
  IF a master record was not found or a
      master record was found and it is
        deleted
    Process unequal records
   IF valid add record
     Indicate updated data obtained
   ELSE
     Obtain valid transaction data              C000
   ENDIF
 ELSE
   Process equal records                        C010
   IF valid change record or valid delete
     record
      Indicate updated data obtained
    ELSE
      Obtain valid transaction data             C000
    ENDIF
  ENDIF
ENDPERFORM
```

FILES, RECORDS, FIELDS REQUIRED

- Updated data obtained indicator
- No more tran records indicator
- Transaction customer number
- Master key area (nominal key)
- Master file input area
  - Delete code field
  - Customer number
  - Customer name
  - Customer address
  - Customer city
  - Customer zip code
  - Customer discount
  - Last activity date
- Valid add indicator
- Valid delete indicator
- Valid change indicator
- Record is found indicator
- Master record being processed work area

HO 2 (cont'd)

Pseudocode Specifications (Part 3 of 12)

Pseudocode Specifications

DATE: June 17	**PROGRAMMER:** Jane Austin	**PROGRAM:** Update Customer Master File
MODULE FUNCTION: Write Updated Customer Master File	**REF:** B010	**MODULE NAME:** Write Updated File

PSEUDOCODE

```
Set valid write indicator to neutral
Move entire master work area to master
    output area
IF the master record being processed =
        key of the record which was retrieved
  Rewrite master record
ELSE
  Write master record
  IF write not successful
    Indicate a valid write did not occur
  ENDIF
ENDIF
```

REF:

FILES, RECORDS, FIELDS REQUIRED

- Valid write indicator
- New master work area
  - Delete code
  - Customer number
  - Customer name
  - Customer address
  - Customer city
  - Customer state
  - Customer zip code
  - Customer discount
  - Last activity date
- Updated master output area
  - Delete code
  - Customer number
  - Customer name
  - Customer address
  - Customer city
  - Customer state
  - Customer zip code
  - Customer discount
  - Last activity date
- Master record being processed work area
- Key of record which was retrieved

Pseudocode Specifications (Part 4 of 12)

Pseudocode Specifications

DATE: June 17	PROGRAMMER: Jane Austin	PROGRAM: Update Customer Master File
MODULE FUNCTION: Obtain Valid Transaction Data	REF: C000	MODULE NAME: Obtain Valid Trans Data

PSEUDOCODE / **REF:**

```
Set valid record indicator to neutral
Obtain transaction record                        D000
PERFORM UNTIL no more transaction
    records or
        valid record(s) are obtained
  IF transaction type 1                          D010
        Edit type 1 transaction                  D020
        Edit type 2 transaction
  ELSE IF transaction type 3                     D030
        Edit type 3 transaction
  ELSE IF transaction type 4                     D040
        Edit type 4 transaction
  ELSE IF transaction type 5
        Move customer number to
          work area
        Edit customer number                     U020
  ENDIF
  IF valid record(s)
        Set valid record indicator to neutral
        Obtain transaction record(s)             D000
  ELSE
        Indicate valid record
  ENDIF
ENDPERFORM
```

FILES, RECORDS, FIELDS REQUIRED

Valid record indicator
No more tran records indicator
Transaction code
"1" constant
"3" constant
"4" constant
"5" constant
Transaction customer number
Customer number work area
  First 4 digits
  Check digit

Pseudocode Specifications (Part 5 of 12)

Pseudocode Specifications

DATE: June 17	PROGRAMMER: Jane Austin	PROGRAM: Update Customer Master File
MODULE FUNCTION: Process Equal Records	REF: C010	MODULE NAME: Process Equal Master and Transaction Records

PSEUDOCODE / **REF:**

```
IF record is a change ("3" or "4")
   Process change transaction                          D050
   Move old master record to new master
       work area
   Indicate valid change
ELSE IF record type is a deletion ("5")
   Move delete code ("FF") to master record
   Move old master record to new master
       work area
   Indicate valid delete
   Move customer number to report work area            U000
   Move "deletion" to report work area
   Write transaction report
ELSE
   Move customer number to report work area
   Move "transaction code" to report work area         U000
   Move "addition of duplicate record" to report
           work area
   Write transaction report
   Indicate no valid change
   Indicate no valid delete
ENDIF
```

FILES, RECORDS, FIELDS REQUIRED

Transaction code
"3" constant
"4" constant
Old master record
  –all fields–
New master work area
  –all fields–
Valid change indicator
"5" constant
Delete code ("FF")
Valid delete indicator
Report work area
  Customer number
  Action taken
  Field in error
  Type of error
"Deletion" constant
Customer number transaction
"Transaction code" constant
"Addition of duplicate rec" constant

Pseudocode Specifications

DATE: June 17	PROGRAMMER: Jane Austin	PROGRAM: Update Customer Master File
MODULE FUNCTION: Process Unequal Transaction Records	REF: C020	MODULE NAME: Process Unequal Records

PSEUDOCODE / **REF:**

```
IF record type is an addition ("1" and "2")
   Move customer number from trans to
       new master  work area
   Move customer name from trans to new master
       work area
   Move customer address from trans to
       new master  work area
   Move customer city from trans to new master
       work area
   Move customer state from trans to new master
       work area
   Move customer zip code from trans to
       new master  work area
   Move customer discount from trans to
       new master  work area
   Move last activity date from trans to
       new master  work area
   Move spaces to delete code
   Move customer number to report work area
   Move "addition" to report work area
   Write transaction report                            U000
   Indicate valid addition
ELSE
   Move customer number to report work area
   Move "transaction code" to report work area
   Move "unmatched master record" to
       report work  area
   Write transaction report                            U000
   Indicate no valid addition
ENDIF
```

FILES, RECORDS, FIELDS REQUIRED

Transaction Records (1 and 2)
  –all fields–
New master work area
  –all fields–
"Addition" constant
Report work area
  Customer number
  Action taken
  Field in error
  Type of error
Valid addition indicator

Pseudocode Specifications (Part 6 of 12)

Pseudocode Specifications

DATE: June 17	PROGRAMMER: Jane Austin	PROGRAM: Update Customer Master File
MODULE FUNCTION: Obtain Valid Transaction Records	REF: D000	MODULE NAME: Obtain Trans Records

PSEUDOCODE — **REF:**

```
Set valid transaction code indicator to neutral
IF transaction file is not sorted
  Sort transaction file                                          E000
  Open sorted transaction file
  Indicate file is sorted
ENDIF
Read a transaction record into first record area
PERFORM UNTIL valid codes or no more trans records
  IF trans code = 3 or 4 or 5
    Move code to current code work area
    Move customer number to current customer
        number work area
    Indicate valid trans code
    IF trans code = 4
      Move record to type 2 work area
    ENDIF
  ELSE IF trans code = 1
    Read a record into second record area
    IF a record was read
      IF trans code = 2 and customer number in first
          record = customer number
          in second record
        Indicate valid trans code
        Move code to work area
        Move customer number to work area
      ELSE
        Move customer number to type 1 to
          report work area
        Move "customer records" to report work area
        Move "type 1 but no type 2" to
          report work area
        Write transaction report                                 U000
        Move second record to first record area
      ENDIF
    ELSE
      Move customer number of type 1 to report work area
      Move "customer records" to report work area
      Move "type 1 but no type 2" to report work area
      Write transaction report                                   U000
    ENDIF
  ELSE IF trans code = 2
    Move customer number to report work area
    Move "customer records" to report work area
    Move "type 2 but no type 1" to report work area
    Write transaction report
    Read a record into first record area
  ELSE
    Move customer number to report work area
    Move "transaction code" to report work area
    Move "invalid record type" to report work area
    Write transaction report                                     U000
    Read a transaction record into first record area
  ENDIF
ENDPERFORM
IF there are no more tran records
  Close sorted transaction file
ENDIF
```

FILES, RECORDS, FIELDS REQUIRED

- Valid trans code indicator
- Transaction file sorted indicator
- Sorted transaction file
- Transaction input area
- First record area
  - Customer number
  - Customer name
  - Customer address
  - Customer city
  - Customer state
  - Customer zip code
  - Transaction code
- Second record area
  - Customer number
  - Customer discount
  - Last activity date
  - Transaction code
- No more tran records indicator
- Record was read indicator
- Report work area
  - Customer number
  - Field in error
  - Type of error
- "Customer records" constant
- "Type 1 but no type 2" constant
- "Type 2 but no type 1" constant
- "Transaction code" constant
- "Invalid record type" constant

Pseudocode Specifications (Part 7 of 12)

Pseudocode Specifications

DATE: June 17	PROGRAMMER: Jane Austin	PROGRAM: Update Customer Master File
MODULE FUNCTION: Edit a Type 3 Transaction	REF: D030	MODULE NAME: Edit Type 3 Transaction

PSEUDOCODE / **REF:**

```
Move customer number to work area
Edit customer number                                  U020
IF state is not = spaces
   Move customer number to work area for
         state code edit
   Move state code to work area
   Edit state code                                    E010
   IF zip codes = spaces
      Move customer number to report work area
      Move "zip code" to report work area
      Move "state changed-zip not" to report
            work area
      Write transaction report                        U000
      Indicate invalid record
   ENDIF
ENDIF
IF zip code not = spaces and zip code
      not numeric
   Move customer number to report work area
   Move "zip code" to report work area
   Move "field is not numeric" to report
         work area
   Write transaction report                           U000
   Indicate invalid record
ENDIF
```

FILES, RECORDS, FIELDS REQUIRED

- Customer number work area
  - First four characters
  - Check digit
- State code – transaction
- State code work area
- Customer number – transaction
- Report work area
  - Customer number
  - Field in error
  - Type of error
- "Zip code" constant
- "State changed – zip not" constant
- "Field not numeric" constant
- Invalid record indicator
- Customer number work area for state code edit

Pseudocode Specifications

DATE: June 17	PROGRAMMER: Jane Austin	PROGRAM: Update Customer Master File
MODULE FUNCTION: Edit a Type 4 Transaction	REF: D040	MODULE NAME: Edit Type 4 Transaction

PSEUDOCODE / **REF:**

```
Move customer number to work area
Edit customer number                                  U020
IF discount is not = spaces
   Move customer number to work area
        for discount edit
   Move discount to work area
   Edit discount                                      E020
ENDIF
IF last activity date not = spaces
   Move customer number to report work area
        for date edit
   Move last activity date to work area
   Edit last activity date                            E030
ENDIF
```

FILES, RECORDS, FIELDS REQUIRED

- Customer number work area
  - First four characters
  - Check digit

- Customer number – transaction
- Discount – transaction
- Discount work area
- Last activity date – transaction
- Last activity date work area

- Customer number work area for discount edit
- Customer number work area for last activity date edit

Pseudocode Specifications (Part 8 of 12)

Pseudocode Specifications

DATE: June 17	PROGRAMMER: Jane Austin	PROGRAM: Update Customer Master File
MODULE FUNCTION: Process a Change Transaction	REF: D050	MODULE NAME: Process Change Transaction

PSEUDOCODE / **REF:**

```
IF transaction code = 3
   Move customer number to type 3 change
      work area
   IF address is not = spaces
      Move transaction address to work area
      Change customer address                    E040
   ENDIF
   IF city is not = spaces
      Move transaction city to work area
      Change customer city                       E050
   ENDIF
   IF state is not = spaces
      Move transaction state to work area
      Change customer state                      E060
   ENDIF
   IF zip code is not = spaces
      Move transaction zip code to work area
      Change customer zip code                   E070
   ENDIF
ELSE
   Move customer number to type 4 change
      work area
   IF discount is not = spaces
      Move transaction discount to work area
      Change customer discount                   E080
   ENDIF
   IF last activity date is not = spaces
      Move transaction last activity date to
         work area
      Change customer last activity date         E090
   ENDIF
ENDIF
```

FILES, RECORDS, FIELDS REQUIRED

Transaction code
"3" constant
Address – transaction
Address work area
City – transaction
City work area
State – transaction
State work area

Zip code – transaction
Zip code work area
Discount – transaction
Discount work area
Last activity date – transaction
Last activity date work area
Customer number – current record
Type 3 change work area for customer number
Type 4 change work area for customer number

Pseudocode Specifications

DATE: June 17	PROGRAMMER: Jane Austin	PROGRAM: Update Customer Master File
MODULE FUNCTION: Sort the Transaction Records	REF: E000	MODULE NAME: Sort Transaction Records

PSEUDOCODE / **REF:**

```
Sort records
   Using unsorted transaction file
   Giving sorted transaction file
```

FILES, RECORDS, FIELDS REQUIRED

Sort work file
Unsorted transaction file
Sorted transaction file

HO 2 (cont'd)

Pseudocode Specifications (Part 9 of 12)

Pseudocode Specifications

DATE: June 17	PROGRAMMER: Jane Austin	PROGRAM: Update Customer Master File
MODULE FUNCTION: Change the Customer Address	REF: E040	MODULE NAME: Change Customer Address

PSEUDOCODE	REF:	FILES, RECORDS, FIELDS REQUIRED
Move master address to report work area		Address – old master
Move transaction address to report work area		Address – new master
Move customer number to report work area		Address--transaction-work
Move "customer address" to report work area		Customer number
Move "change" to report work area		Report work area
Write transaction report	U000	Customer number
Move transaction address to master address		Action taken
		Field changed
		Old master
		New master
		"Customer state" constant
		"Change" constant

Pseudocode Specifications

DATE: June 17	PROGRAMMER: Jane Austin	PROGRAM: Update Customer Master File
MODULE FUNCTION: Change the Customer City	REF: E050	MODULE NAME: Change Customer City

PSEUDOCODE	REF:	FILES, RECORDS, FIELDS REQUIRED
Move master city to report work area		City – old master
Move transaction city to report work		City – new master
Move customer number to report work area		City – transaction-work
Move "customer city" to report work area		Customer number
Move "change" to report work area		Report work area
Write transaction report	U000	Customer number
Move transaction city to master city		Action taken
		Field changed
		Old master
		New master
		"Last activity date" constant
		"Change" constant

Pseudocode Specifications (Part 10 of 12)

Pseudocode Specifications

DATE: June 17	PROGRAMMER: Jane Austin	PROGRAM: Update Customer Master File
MODULE FUNCTION: Change the Customer State	REF: E060	MODULE NAME: Change Customer State

PSEUDOCODE — **REF:**

```
Move master state to report work area
Move transaction state to report work area
Move customer number to report work area
Move "customer state" to report work area
Move "change" to report work area
Write transaction report                          U000
Move transaction state to master state
```

FILES, RECORDS, FIELDS REQUIRED

- State – old master
- State – new master
- State –transaction-work
- Customer number
- Report work area
    - Customer number
    - Action taken
    - Field changed
    - Old master
    - New master
- "Customer state" constant
- "Change" constant

Pseudocode Specifications

DATE: June 17	PROGRAMMER: Jane Austin	PROGRAM: Update Customer Master File
MODULE FUNCTION: Change the Customer Zip Code	REF: E070	MODULE NAME: Change Customer Zip Code

PSEUDOCODE — **REF:**

```
Move master zip code to report work area
Move transaction zip code to report
    work area
Move customer number to report work area
Move "customer zip code" to report
    work area
Move "Change" to report work area                 U000
Write transaction report
Move transaction zip code to master
    zip code
```

FILES, RECORDS, FIELDS REQUIRED

- Zip code – old master
- Zip code – new master
- Zip code– transaction-work
- Customer number
- Report work area
    - Customer number
    - Action taken
    - Field changed
    - Old master
    - New master
- "Customer zip code" constant
- "Change" constant

Pseudocode Specifications (Part 11 of 12)

Pseudocode Specifications

DATE: June 17
PROGRAMMER: Jane Austin
PROGRAM: Update Customer Master File
MODULE FUNCTION: Change the Last Activity Date
REF: E090
MODULE NAME: Change Last Activity Date

PSEUDOCODE

```
Move master last activity date to report work area
Move transaction last activity date to report
    work area
Move customer number to report work area
Move "last activity date" to report work area
Move "change" to report work area
Write transaction report
Move transaction last activity date to
        master last activity date
```

REF:

FILES, RECORDS, FIELDS REQUIRED

Last activity date – old master
Last activity date – new master
Last activity date – transaction-work
Customer number
Report work area
- Customer number
- Action taken
- Field changed
- Old master
- New master

"Last activity date" constant
"Change" constant

Pseudocode Specifications

DATE: June 17
PROGRAMMER: Jane Austin
PROGRAM: Update Customer Master File
MODULE FUNCTION: Change the Customer Discount
REF: E080
MODULE NAME: Change Customer Discount

PSEUDOCODE	REF:
Move master discount to report work area	
Move transaction discount to report work area	
Move customer number to report work area	
Move "customer discount" to report work area	
Move "change" to report work area	
Write transaction report	U000
Move transaction discount to master discount	

FILES, RECORDS, FIELDS REQUIRED

Discount – old master
Discount – new master
Discount – transaction-work
Customer number
Report work area
- Customer number
- Action taken
- Field changed
- Old master
- New master

"Customer discount" constant
"Change" constant

Pseudocode Specifications (Part 12 of 12)

Pseudocode Specifications

DATE: June 17	**PROGRAMMER:** Jane Austin	**PROGRAM:** Update Customer Master File
MODULE FUNCTION: Write the Transaction Report	**REF:** U000	**MODULE NAME:** Write Transaction Report

PSEUDOCODE | **REF:**

```
IF first page or line count> page size
  Print headings                                  U010
  Clear report area
  Move customer number to report area
ENDIF
IF current customer number not = previous
    customer number
  Move customer number to report area
  Move customer number to compare area
  Set spacing for double spacing
ENDIF
Move action taken to report area
Move field to report area
Move type of error to report area
Move old master data to report area
Move new master data to report area
Write a line on the report
Add spacing to number of lines printed
Set spacing for single spacing
Clear report area
Clear report work area
```

FILES, RECORDS, FIELDS REQUIRED

First page indicator
Line count
Page size
Report output area
    Customer number
    Action taken
    Field changed or in error
    Type of error
    Old master record
    New master record
Report work area
    Customer number
    Action taken
    Field changed or in error
    Type of error
    Old master record
    New master record
Previous customer compare area
Spacing control field
Double spacing control character
Single spacing control character

Pseudocode Specifications

DATE: June 17	**PROGRAMMER:** Jane Austin	**PROGRAM:** Update Customer Master File
MODULE FUNCTION: Print Transaction Report Headings	**REF:** U010	**MODULE NAME:** Print Headings

PSEUDOCODE | **REF:**

```
Move current date to first heading line
Move page number to first heading line
Write first heading line
Add 1 to page number
Write second heading line
Write third heading line
Set spacing for double spacing
Set line count to zero
```

FILES, RECORDS, FIELDS REQUIRED

Current date
First heading line
    Current date
    Page number
Page number
Second heading line
Third heading line
Spacing control field
Double spacing control character
Line count

Source Listing

(Part 1 of 7)

```
001010 IDENTIFICATION DIVISION.
001020
001030 PROGRAM-ID.     CUSTUPDT.
001040 AUTHOR.         SHELLY AND CASHMAN.
001050 INSTALLATION.   ANAHEIM.
001060 DATE-WRITTEN.   07/27/78.
001070 DATE-COMPILED.  SEP  5,1978.
001080 SECURITY.       UNCLASSIFIED.
001090
001100*****************************************************************
001110*                                                               *
001120*  THIS PROGRAM UPDATES THE CUSTOMER MASTER FILE, AN INDEXED    *
001130*  SEQUENTIAL FILE. RECORDS CAN BE ADDED TO THE FILE, RECORDS   *
001140*  CAN BE DELETED FROM THE FILE, AND FIELDS WITHIN A MASTER     *
001150*  RECORD CAN BE CHANGED. THE TRANSACTION FILE RECORDS MUST BE  *
001160*  SORTED IN THE PROGRAM BECAUSE AN ADDITION TO THE MASTER FILE *
001170*  REQUIRES A TYPE 1 AND TYPE 2 TRANSACTION IN SEQUENCE         *
001180*  WITHIN CUSTOMER NUMBER. EACH TRANSACTION WHICH IS READ       *
001190*  WILL BE COMPLETELY EDITED BEFORE BEING USED TO UPDATE THE    *
001200*  CUSTOMER MASTER FILE. A TRANSACTION REPORT IS PRINTED WHICH  *
002010*  CONTAINS ANY ERRORS FOUND IN EDITING AND ALSO A LIST OF ALL  *
002020*  ADDITIONS, CHANGES, OR DELETIONS WHICH ARE PERFORMED ON      *
002030*  THE CUSTOMER MASTER FILE.                                   .*
002040*                                                               *
002050*****************************************************************
002060
002070
002080
002090 ENVIRONMENT DIVISION.
002100
002110 CONFIGURATION SECTION.
002120
002130 SOURCE-COMPUTER. IBM-370.
002140 OBJECT-COMPUTER. IBM-370.
002150 SPECIAL-NAMES.   C01 IS TO-THE-TOP-OF-THE-PAGE.
002160
002170 INPUT-OUTPUT SECTION.
002180
002190 FILE-CONTROL.
002200     SELECT CUSTOMER-TRANS-INPUT-FILE
003010         ASSIGN TO UR-S-SYSIN.
003020     SELECT SORTED-CUST-TRANS-INPUT-FILE
003030         ASSIGN TO DA-S-SORTRANS.
003040     SELECT CUSTOMER-MASTER-I-O-FILE
003050         ASSIGN TO DA-I-CUSTMAST
003060         ACCESS IS RANDOM
003070         RECORD KEY IS CUSTOMER-NUMBER-MSTRIO
003080         NOMINAL KEY IS KEY-OF-THE-RECORD-TO-RETRIEVE.
003090     SELECT UPDATE-TRANSACTION-REPORT-FILE
003100         ASSIGN TO UR-S-SYSOUT.
003110     SELECT TRANSACTION-SORT-WORK-FILE
003120         ASSIGN TO DA-S-SORTWK01.
003140 DATA DIVISION.
003150
003160 FILE SECTION.
003170
003180 FD  CUSTOMER-TRANS-INPUT-FILE
003190       RECORD CONTAINS 80 CHARACTERS
003200       LABEL RECORDS ARE OMITTED
004010       DATA RECORD IS TRANSACTION-INPUT-RECORD.
004020
004030 01  TRANSACTION-INPUT-RECORD          PIC X(80).
004040
004050 FD  SORTED-CUST-TRANS-INPUT-FILE
004060       BLOCK CONTAINS 20 RECORDS
004070       RECORD CONTAINS 80 CHARACTERS
004080       LABEL RECORDS ARE STANDARD
004090       DATA RECORD IS SORTED-CUST-TRANS-INPUT-RECORD.
004100
004110 01  SORTED-CUST-TRANS-INPUT-RECORD  PIC X(80).
004120
004130 FD  CUSTOMER-MASTER-I-O-FILE
004140       BLOCK CONTAINS 20 RECORDS
004150       RECORD CONTAINS 88 CHARACTERS
004160       LABEL RECORDS ARE STANDARD
004170       DATA RECORD IS CUSTOMER-MASTER-I-O-RECORD.
004180
004190 01  CUSTOMER-MASTER-I-O-RECORD.
004200     05  FILLER                       PIC X.
004210     05  CUSTOMER-NUMBER-MSTRIO       PIC X(5).
004220     05  FILLER                       PIC X(82).
004230
005010 FD  UPDATE-TRANSACTION-REPORT-FILE
005020       RECORD CONTAINS 133 CHARACTERS
005030       LABEL RECORDS ARE OMITTED
005040       DATA RECORD IS UPDATE-TRANSACTION-REPORT-LINE.
005050
005060 01  UPDATE-TRANSACTION-REPORT-LINE.
005070     05  CARRIAGE-CONTROL             PIC X.
005080     05  FILLER                       PIC X.
005090     05  CUSTOMER-NUMBER-REPORT       PIC X(5).
005100     05  FILLER                       PIC X(4).
005110     05  ACTION-TAKEN-REPORT          PIC X(8).
005120     05  FILLER                       PIC XX.
005130     05  FIELD-CHANGED-ERROR-REPORT   PIC X(20).
005140     05  FILLER                       PIC XX.
005150     05  TYPE-OF-ERROR-REPORT         PIC X(25).
005160     05  FILLER                       PIC XX.
005170     05  OLD-MASTER-RECORD-REPORT     PIC X(27).
005180     05  FILLER                       PIC XX.
005190     05  NEW-MASTER-RECORD-REPORT     PIC X(27).
005200     05  FILLER                       PIC X(7).
006010
006020 SD  TRANSACTION-SORT-WORK-FILE
006030       COPY SORTSD.
C 008020       RECORD CONTAINS 80 CHARACTERS
C 008030       DATA RECORD IS TRANSACTION-SORT-WORK-RECORD.
C 008040
C 008050 01  TRANSACTION-SORT-WORK-RECORD.
C 009010     05  CUSTOMER-NUMBER-SORTREC      PIC X(5).
C 009020     05  FILLER                       PIC X(74).
C 009030     05  TRANSACTION-CODE-SORTREC     PIC X.
**006040
  006050 WORKING-STORAGE SECTION.
  006060
  006070 01  PROGRAM-INDICATORS.
  006080     05  ARE-THERE-MORE-TRAN-RECORDS     PIC XXX VALUE 'YES'.
  006090         88  THERE-ARE-MORE-TRAN-RECORDS         VALUE 'YES'.
  006100         88  A-TRAN-RECORD-WAS-READ              VALUE 'YES'.
  006110         88  THERE-ARE-NO-MORE-TRAN-RECORDS      VALUE 'NO '.
  006120     05  ARE-TRAN-RECORDS-VALID          PIC XXX.
  006130         88  TRAN-RECORDS-ARE-VALID              VALUE 'YES'.
  006140         88  TRAN-RECORDS-ARE-NOT-VALID          VALUE 'NO '.
  006150     05  HAS-UPDATED-DATA-BEEN-OBTAINED  PIC XXX.
  006160         88  UPDATED-DATA-HAS-BEEN-OBTAINED      VALUE 'YES'.
  006170         88  DATA-HAS-NOT-BEEN-OBTAINED          VALUE 'NO '.
  006180     05  WAS-IT-A-VALID-ADD-RECORD       PIC XXX.
  006190         88  IT-WAS-A-VALID-ADD-RECORD           VALUE 'YES'.
  007010     05  WAS-IT-A-VALID-DELETE-RECORD    PIC XXX.
  007020         88  IT-WAS-A-VALID-DELETE-RECORD        VALUE 'YES'.
  007030     05  WAS-IT-A-VALID-CHANGE-RECORD    PIC XXX.
  007040         88  IT-WAS-A-VALID-CHANGE-RECORD        VALUE 'YES'.
  007050     05  WAS-A-MASTER-RECORD-FOUND       PIC XXX.
  007060         88  A-MASTER-RECORD-WAS-FOUND           VALUE 'YES'.
  007070         88  A-MASTER-RECORD-WAS-NOT-FOUND       VALUE 'NO '.
  007080     05  DID-A-VALID-WRITE-OCCUR         PIC XXX.
  007090         88  A-VALID-WRITE-DID-NOT-OCCUR         VALUE 'NO '.
  007100     05  IS-TRANSACTION-CODE-VALID       PIC XXX.
  007110         88  TRANSACTION-CODE-IS-VALID           VALUE 'YES'.
  007120         88  TRANSACTION-CODE-IS-NOT-VALID       VALUE 'NO '.
  007130     05  IS-TRANSACTION-FILE-SORTED      PIC XXX VALUE 'NO '.
  007140         88  TRANSACTION-FILE-IS-NOT-SORTED      VALUE 'NO '.
  007150
  007160 01  PROGRAM-COMPARE-AREAS.
  007170     05  PREVIOUS-CUSTOMER-NUMBER    PIC X(5) VALUE LOW-VALUES.
  007180
  007190 01  NOMINAL-KEY-FIELD.
  007200     05  KEY-OF-THE-RECORD-TO-RETRIEVE   PIC X(5).
  008010
  008020 01  PROGRAM-WORK-AREAS.
  008030     05  TRANSACTION-REPORT-WORK-AREA.
  008040         10  CUSTOMER-NUMBER-RPTWORK         PIC X(5)  VALUE
  008050                                                       SPACES.
  008060         10  ACTION-TAKEN-RPTWORK            PIC X(8)  VALUE
  008070                                                       SPACES.
  008080         10  FIELD-CHANGED-ERROR-RPTWORK     PIC X(20) VALUE
  008090                                                       SPACES.
  008100         10  TYPE-OF-ERROR-RPTWORK           PIC X(25) VALUE
  008110                                                       SPACES.
  008120         10  OLD-MASTER-RECORD-RPTWORK       PIC X(27) VALUE
  008130                                                       SPACES.
  008140         10  NEW-MASTER-RECORD-RPTWORK       PIC X(27) VALUE
  008150                                                       SPACES.
  008160     05  CUSTOMER-NUMBER-WORK.
  008170         10  FIRST-FOUR-CHAR-CUST-NUMB-WORK  PIC X(4).
  008180         10  FIFTH-CHAR-CUST-NUMB-WORK       PIC X.
  008190             88  VALID-CHECK-DIGIT-FOR-TEN    VALUE 'X'.
  008200     05  STATE-EDIT-WORK                     PIC XX.
  008210     05  DISCOUNT-EDIT-WORK                  PIC XX.
  008220     05  LAST-ACTIVITY-EDIT-WORK.
  008230         10  MONTH-LAST-ACTIVITY-WORK        PIC 99.
  008240             88  VALID-MONTH-WORK             VALUE 01 THRU 12.
  008250         10  DAY-LAST-ACTIVITY-WORK          PIC XX.
  008260         10  YEAR-LAST-ACTIVITY-WORK         PIC XX.
  008270     05  ADDRESS-CHANGE-WORK                 PIC X(27).
  008280     05  CITY-CHANGE-WORK                    PIC X(20).
  008290     05  STATE-CHANGE-WORK                   PIC XX.
  008300     05  ZIP-CODE-CHANGE-WORK                PIC X(5).
  008310     05  DISCOUNT-CHANGE-WORK                PIC XX.
  008320     05  LAST-ACTIVITY-CHANGE-WORK           PIC X(6).
  008330     05  CUSTOMER-NUMBER-STATEDIT-WORK       PIC X(5).
  008340     05  CUSTOMER-NUMBER-DISCEDIT-WORK       PIC X(5).
  008350     05  CUSTOMER-NUMBER-DATEDIT-WORK        PIC X(5).
  009010     05  CUSTOMER-NUMBER-TYPE-3-CHANGE       PIC X(5).
  009020     05  CUSTOMER-NUMBER-TYPE-4-CHANGE       PIC X(5).
  009030     05  TRANSACTION-CODE-CURRENT-REC        PIC X.
  009040     05  CUSTOMER-NUMBER-CURRENT-REC         PIC X(5).
  009050     05  MASTER-RECORD-BEING-PROCESSED       PIC X(5).
  009060
  009070 01  MOD-11-PASS-AREA-WORK.
  009080     05  FIELD-TO-BE-CHECKED-MOD11-WORK      PIC X(20).
  009090     05  DID-CUST-NUMBER-PASS-MOD11          PIC XXX.
  009100         88  CUST-NUMBER-PASSED-MOD11         VALUE 'YES'.
  009110         88  CUST-NUMBER-DID-NOT-PASS-MOD11   VALUE 'NO '.
  009120
  009130 01  PROGRAM-CONSTANTS
  009140       COPY PGMCON.
C 011060 01  PROGRAM-CONSTANTS.
C 011070     05  CUSTOMER-NAME-FIELD-MSG               PIC X(20) VALUE
C 011080                                            'CUSTOMER NAME
C 011090     05  BLANK-FIELD-ERROR-MSG                 PIC X(25) VALUE
C 011100                                         'FIELD CONTAINS SPACES
C 011110     05  CUSTOMER-ADDRESS-FIELD-MSG            PIC X(20) VALUE
C 011120                                            'CUSTOMER ADDRESS
C 011130     05  CUSTOMER-CITY-FIELD-MSG               PIC X(20) VALUE
C 011140                                            'CUSTOMER CITY
C 011150     05  CUSTOMER-STATE-FIELD-MSG              PIC X(20) VALUE
C 011160                                            'CUSTOMER STATE
C 011170     05  BAD-STATE-CODE-ERROR-MSG              PIC X(25) VALUE
C 011180                                           'INVALID STATE CODE
C 011190     05  CUSTOMER-ZIP-CODE-FIELD-MSG           PIC X(20) VALUE
```

Source Listing (Part 2 of 7)

```
C 011200                                                'CUSTOMER ZIP CODE
C 012010     05  FIELD-NOT-NUMERIC-ERROR-MSG                PIC X(25) VALUE
C 012020                                                'FIELD IS NOT NUMERIC
C 012030     05  CUSTOMER-DISCOUNT-FIELD-MSG                PIC X(20) VALUE
C 012040                                                    'CUSTOMER DISCOUNT
C 012050     05  MAXIMUM-DISCOUNT-PERCENT                   PIC XX     VALUE '20
C 012060     05  BAD-DISCOUNT-ERROR-MSG                     PIC X(25) VALUE
C 012070                                                'DISCOUNT EXCEEDS MAXIMUM
C 012080     05  LAST-ACTIVITY-DATE-FIELD-MSG               PIC X(20) VALUE
C 012090                                                    'LAST ACTIVITY DATE
C 012120     05  BAD-MONTH-ERROR-MSG                        PIC X(25) VALUE
C 012130                                                'INVALID MONTH
C 012140     05  FIRST-DAY-OF-MONTH-NUMBER                  PIC XX     VALUE '01
C 012150     05  BAD-DAY-ERROR-MSG                          PIC X(25) VALUE
C 012160                                                'INVALID DAY
C 012170     05  CUSTOMER-RECORDS-FIELD-MSG                 PIC X(20) VALUE
C 012180                                                    'CUSTOMER RECORDS
C 012190     05  TYPE1-BUT-NO-TYPE2-ERROR-MSG               PIC X(25) VALUE
C 012200                                                'TYPE 1 BUT NO TYPE 2
C 012210     05  TRANSACTION-TYPE-ONE                       PIC X      VALUE '1'.
C 012220     05  TRANSACTION-TYPE-TWO                       PIC X      VALUE '2'.
C 013010     05  TYPE2-BUT-NO-TYPE1-ERROR-MSG               PIC X(25) VALUE
C 013020                                                'TYPE 2 BUT NO TYPE 1
C 013030     05  BAD-TRANS-CODE-ERROR-MSG                   PIC X(25) VALUE
C 013040                                                'INVALID RECORD TYPE
C 013050     05  CUSTOMER-NUMBER-FIELD-MSG                  PIC X(20) VALUE
C 013060                                                    'CUSTOMER NUMBER
C 013070     05  FAILS-MOD11-ERROR-MSG                      PIC X(25) VALUE
C 013080                                                'FAILS MOD-11 CHECK
C 013090     05  INVALID-WRITE-MSG                          PIC X(25) VALUE
C 013100                                                'DUPLICATE OR BAD SEQUENCE'
**009150     05  DELETE-CODE-CONSTANT                       PIC X      VALUE
  009160                                                     HIGH-VALUES.
  009170     05  TRANSACTION-TYPE-THREE                     PIC X      VALUE '3'.
  009180     05  TRANSACTION-TYPE-FOUR                      PIC X      VALUE '4'.
  009190     05  TRANSACTION-TYPE-FIVE                      PIC X      VALUE '5'.
  009200     05  DELETION-MSG                               PIC X(8)  VALUE
  009210                                                              'DELETION'.
  009220     05  TRANSACTION-CODE-FIELD-MSG                 PIC X(20) VALUE
  009230                                                    'TRANSACTION CODE
  009240     05  ADD-OF-DUPLICATE-ERROR-MSG                 PIC X(25) VALUE
  009250                                                'ADDITION OF DUPLICATE REC
  009260     05  ADDITION-MSG                               PIC X(8)  VALUE
  009270                                                              'ADDITION'
  009280     05  STATE-BUT-NO-ZIP-ERROR-MSG                 PIC X(25) VALUE
  009290                                                'STATE CHANGED - ZIP NOT
  009300     05  CHANGE-MSG                                 PIC X(8)  VALUE
  009310                                                              'CHANGE
  010010     05  NO-MASTER-RECORD-MSG                       PIC X(25) VALUE
  010020                                                'NO MASTER RECORD
  010070
  010080 01  PROGRAM-TABLES
  010090       COPY PGMTBLS.
C 013120 01  PROGRAM-TABLES.
C 013130     05  STATE-TABLE.
C 013140         10  VALID-STATE-CONSTANTS.
C 013150             15  FILLER             PIC XX         VALUE 'AK'.
C 013160             15  FILLER             PIC XX         VALUE 'AL'.
C 013170             15  FILLER             PIC XX         VALUE 'AR'.
C 013180             15  FILLER             PIC XX         VALUE 'AZ'.
C 013190             15  FILLER             PIC XX         VALUE 'CA'.
C 013200             15  FILLER             PIC XX         VALUE 'CO'.
C 013210             15  FILLER             PIC XX         VALUE 'CT'.
C 013220             15  FILLER             PIC XX         VALUE 'DC'.
C 013230             15  FILLER             PIC XX         VALUE 'DE'.
C 014010             15  FILLER             PIC XX         VALUE 'FL'.
C 014020             15  FILLER             PIC XX         VALUE 'GA'.
C 014030             15  FILLER             PIC XX         VALUE 'HI'.
C 014040             15  FILLER             PIC XX         VALUE 'IA'.
C 014050             15  FILLER             PIC XX         VALUE 'ID'.
C 014060             15  FILLER             PIC XX         VALUE 'IL'.
C 014070             15  FILLER             PIC XX         VALUE 'IN'.
C 014080             15  FILLER             PIC XX         VALUE 'KS'.
C 014090             15  FILLER             PIC XX         VALUE 'KY'.
C 014100             15  FILLER             PIC XX         VALUE 'LA'.
C 014110             15  FILLER             PIC XX         VALUE 'MA'.
C 014120             15  FILLER             PIC XX         VALUE 'MD'.
C 014130             15  FILLER             PIC XX         VALUE 'ME'.
C 014140             15  FILLER             PIC XX         VALUE 'MI'.
C 014150             15  FILLER             PIC XX         VALUE 'MN'.
C 014160             15  FILLER             PIC XX         VALUE 'MO'.
C 014170             15  FILLER             PIC XX         VALUE 'MS'.
C 014180             15  FILLER             PIC XX         VALUE 'MT'.
C 014190             15  FILLER             PIC XX         VALUE 'NB'.
C 014200             15  FILLER             PIC XX         VALUE 'NC'.
C 015010             15  FILLER             PIC XX         VALUE 'ND'.
C 015020             15  FILLER             PIC XX         VALUE 'NH'.
C 015030             15  FILLER             PIC XX         VALUE 'NJ'.
C 015040             15  FILLER             PIC XX         VALUE 'NM'.
C 015050             15  FILLER             PIC XX         VALUE 'NV'.
C 015060             15  FILLER             PIC XX         VALUE 'NY'.
C 015070             15  FILLER             PIC XX         VALUE 'OH'.
C 015080             15  FILLER             PIC XX         VALUE 'OK'.
C 015090             15  FILLER             PIC XX         VALUE 'OR'.
C 015100             15  FILLER             PIC XX         VALUE 'PA'.
C 015110             15  FILLER             PIC XX         VALUE 'RI'.
C 015120             15  FILLER             PIC XX         VALUE 'SC'.
C 015130             15  FILLER             PIC XX         VALUE 'SD'.
C 015140             15  FILLER             PIC XX         VALUE 'TN'.
C 015150             15  FILLER             PIC XX         VALUE 'TX'.
C 015160             15  FILLER             PIC XX         VALUE 'UT'.
C 015170             15  FILLER             PIC XX         VALUE 'VA'.
C 015180             15  FILLER             PIC XX         VALUE 'VT'.
C 015190             15  FILLER             PIC XX         VALUE 'WA'.
C 015200             15  FILLER             PIC XX         VALUE 'WI'.
C 016010             15  FILLER             PIC XX         VALUE 'WV'.
C 016020             15  FILLER             PIC XX         VALUE 'WY'.
C 016030         10  VALID-STATE-TABLE REDEFINES VALID-STATE-CONSTANTS
C 016040                                             OCCURS 51 TIMES
C 016050                                             ASCENDING KEY IS
C 016060                                                 STATE-CODE
C 016070                                             INDEXED BY
C 016080                                                 STATE-CODE-INDEX
```

```
  C 016090             15  STATE-CODE         PIC XX.
  C 016100     05  MONTH-DAY-TABLE.
  C 016110         10  MONTH-DAY-CONSTANTS.
  C 016120             15  FILLER                  PIC X(4)       VALUE '0131'.
  C 016130             15  FILLER                  PIC X(4)       VALUE '0229'.
  C 016140             15  FILLER                  PIC X(4)       VALUE '0331'.
  C 016150             15  FILLER                  PIC X(4)       VALUE '0430'.
  C 016160             15  FILLER                  PIC X(4)       VALUE '0531'.
  C 016170             15  FILLER                  PIC X(4)       VALUE '0630'.
  C 016180             15  FILLER                  PIC X(4)       VALUE '0731'.
  C 016190             15  FILLER                  PIC X(4)       VALUE '0831'.
  C 016200             15  FILLER                  PIC X(4)       VALUE '0930'.
  C 017010             15  FILLER                  PIC X(4)       VALUE '1031'.
  C 017020             15  FILLER                  PIC X(4)       VALUE '1130'.
  C 017030             15  FILLER                  PIC X(4)       VALUE '1231'.
  C 017040         10  MONTHS-AND-DAYS-TABLE REDEFINES MONTH-DAY-CONSTANTS
  C 017050                                             OCCURS 12 TIMES
  C 017060                                             INDEXED BY
  C 017070                                                MONTH-DAY-INDEX.
  C 017080             15  MONTH-TABLE        PIC XX.
  C 017090             15  NUMBER-OF-DAYS-IN-MONTH PIC XX.
  **010100
*   010110 01  TRANSACTION-RECORDS-WORK-AREA
    010120       COPY INPRECS.
* C 017110 01  TRANSACTION-RECORDS-WORK-AREA.
  C 017120     05  SORTED-TRANS-RECORD-TYPE-1.
* C 017130         10  CUSTOMER-NUMBER-TRANSORT1               PIC X(5).
  C 017140         10  CUSTOMER-NAME-TRANSORT1                 PIC X(20).
* C 017150         10  CUSTOMER-ADDRESS-TRANSORT1              PIC X(27).
  C 017160         10  CUSTOMER-CITY-TRANSORT1                 PIC X(20).
  C 017170         10 CUSTOMER-STATE-TRANSORT1                 PIC XX.
  C 017180         10 CUSTOMER-ZIP-CODE-TRANSORT1              PIC X(5).
  C 017190         10  TRANSACTION-CODE-TRANSORT1              PIC X.
  C 017200     05  SORTED-TRANS-RECORD-TYPE-2.
  C 017210         10  CUSTOMER-NUMBER-TRANSORT2               PIC X(5).
  C 017220         10  CUSTOMER-DISCOUNT-TRANSORT2             PIC XX.
  C 017230         10  LAST-ACTIVITY-DATE-TRANSORT2.
  C 017240             15  MONTH-LAST-ACTIVITY-TRANSORT2       PIC 99.
  C 017250                 88  VALID-MONTH                     VALUE 01 THRU 12
  C 017260             15  DAY-LAST-ACTIVITY-TRANSORT2         PIC XX.
  C 017270             15  YEAR-LAST-ACTIVITY-TRANSORT2        PIC XX.
  C 017280         10  FILLER                                  PIC X(66).
  C 017290         10  TRANSACTION-CODE-TRANSORT2              PIC X.
  **010130
    010140 01  MASTER-INPUT-WORK-AREA.
    010150     05  DELETE-CODE-MSTRIN-WORK           PIC X.
    010160         88  CUSTOMER-RECORD-IS-DELETED        VALUE HIGH-VALUES.
    010170     05  CUSTOMER-NUMBER-MSTRIN-WORK       PIC X(5).
    010180     05  CUSTOMER-NAME-MSTRIN-WORK         PIC X(20).
    010190     05  CUSTOMER-ADDRESS-MSTRIN-WORK      PIC X(27).
    010200     05  CUSTOMER-CITY-MSTRIN-WORK         PIC X(20).
    011010     05  CUSTOMER-STATE-MSTRIN-WORK        PIC XX.
    011020     05  CUSTOMER-ZIP-CODE-MSTRIN-WORK     PIC X(5).
    011030     05  CUSTOMER-DISCOUNT-MSTRIN-WORK     PIC XX.
    011040     05  LAST-ACTIVITY-MSTRIN-WORK         PIC X(6).
    011050
    011060 01  MASTER-OUTPUT-WORK-AREA.
    011070     05  DELETE-CODE-MSTROUT-WORK          PIC X.
    011080     05  CUSTOMER-NUMBER-MSTROUT-WORK      PIC X(5).
    011090     05  CUSTOMER-NAME-MSTROUT-WORK        PIC X(20).
    011100     05  CUSTOMER-ADDRESS-MSTROUT-WORK     PIC X(27).
    011110     05  CUSTOMER-CITY-MSTROUT-WORK        PIC X(20).
    011120     05  CUSTOMER-STATE-MSTROUT-WORK       PIC XX.
    011130     05  CUSTOMER-ZIP-CODE-MSTROUT-WORK    PIC X(5).
    011140     05  CUSTOMER-DISCOUNT-MSTROUT-WORK    PIC XX.
    011150     05  LAST-ACTIVITY-MSTROUT-WORK        PIC X(6).
    011160
    011170 01  PRINTER-CONTROL.
    011180     05  PROPER-SPACING                PIC 9.
    011190     05  SPACE-ONE-LINE                PIC 9     VALUE 1.
    011200     05  SPACE-TWO-LINES               PIC 9     VALUE 2.
    012010     05  NUMBER-OF-LINES-PRINTED       PIC S999  VALUE ZERO
    012020                                               USAGE IS COMP-3.
    012030     05  PAGE-SIZE                     PIC 999   VALUE 40
    012040                                               USAGE IS COMP-3.
    012050     05  PAGE-NUMBER-COUNT             PIC S999  VALUE +1
    012060                                               USAGE IS COMP-3.
    012070         88  THIS-IS-THE-FIRST-PAGE            VALUE +1.
    012080
    012090 01  HEADING-LINES.
    012100     05  FIRST-HEADING-LINE.
    012110         10  CARRIAGE-CONTROL      PIC X.
    012120         10  FILLER                PIC X(5)      VALUE 'DATE '.
    012130         10  DATE-HDG1             PIC X(8).
    012140         10  FILLER                PIC X(37)     VALUE SPACES.
    012150         10  FILLER                PIC X(7)      VALUE 'UPDATE '.
    012160         10  FILLER                PIC X(12)     VALUE
    012170                                                 'TRANSACTION '.
    012180         10  FILLER                PIC X(6)      VALUE 'REPORT'.
    012190         10  FILLER                PIC X(42)     VALUE SPACES.
    012200         10  FILLER                PIC X(5)      VALUE 'PAGE '.
    013010         10  PAGE-NUMBER-HDG1      PIC ZZ9.
    013020         10  FILLER                PIC X(7)      VALUE SPACES.
    013030
    013040     05  SECOND-HEADING-LINE.
    013050         10  CARRIAGE-CONTROL      PIC X.
    013060         10  FILLER                PIC X(8)      VALUE 'CUSTOMER'.
    013070         10  FILLER                PIC XXX       VALUE SPACES.
    013080         10  FILLER                PIC X(6)      VALUE 'ACTION'.
    013090         10  FILLER                PIC X(5)      VALUE SPACES.
    013100         10  FILLER                PIC X(6)      VALUE 'FIELD '.
    013110         10  FILLER                PIC X(10)     VALUE 'CHANGED OR'
    013120         10  FILLER                PIC X(94)     VALUE SPACES.
    013130
    013140     05  THIRD-HEADING-LINE.
    013150         10  CARRIAGE-CONTROL      PIC X.
    013160         10  FILLER                PIC X         VALUE SPACE.
    013170         10  FILLER                PIC X(6)      VALUE 'NUMBER'.
    013180         10  FILLER                PIC X(4)      VALUE SPACES.
    013190         10  FILLER                PIC X(5)      VALUE 'TAKEN'.
    013200         10  FILLER                PIC X(7)      VALUE SPACES.
    014010         10  FILLER                PIC X(6)      VALUE 'FIELD '.
    014020         10  FILLER                PIC X(8)      VALUE 'IN ERROR'.
```

BEST COPY

Source Listing (Part 3 of 7)

```
           10  FILLER                  PIC X(11)   VALUE SPACES.
           10  FILLER                  PIC X(5)    VALUE 'TYPE '.
           10  FILLER                  PIC X(8)    VALUE 'OF ERROR'.
           10  FILLER                  PIC X(12)   VALUE SPACES.
           10  FILLER                  PIC X(11)   VALUE 'OLD MASTER
           10  FILLER                  PIC X(6)    VALUE 'RECORD'.
           10  FILLER                  PIC X(12)   VALUE SPACES.
           10  FILLER                  PIC X(11)   VALUE 'NEW MASTER
           10  FILLER                  PIC X(6)    VALUE 'RECORD'.
           10  FILLER                  PIC X(13)   VALUE SPACES.
 PROCEDURE DIVISION.

*****************************************************************
*                                                               *
*  THE FUNCTION OF THIS MODULE IS TO UPDATE THE CUSTOMER        *
*  MASTER FILE. IT IS ENTERED FROM AND EXITS TO THE OPERATING   *
*  SYSTEM.                                                      *
*                                                               *
*****************************************************************

 A000-UPDATE-CUSTOMER-MASTER.

     OPEN I-O    CUSTOMER-MASTER-I-O-FILE
          OUTPUT UPDATE-TRANSACTION-REPORT-FILE.
     PERFORM B000-OBTAIN-UPDATED-DATA.
     PERFORM A001-WRITE-AND-READ
          UNTIL THERE-ARE-NO-MORE-TRAN-RECORDS.
     CLOSE CUSTOMER-MASTER-I-O-FILE
          UPDATE-TRANSACTION-REPORT-FILE.
     STOP RUN.



 A001-WRITE-AND-READ.

     PERFORM B010-WRITE-UPDATED-FILE.
     PERFORM B000-OBTAIN-UPDATED-DATA.
*****************************************************************
*                                                               *
*  THIS MODULE OBTAINS UPDATED DATA FOR THE CUSTOMER MASTER     *
*  FILE UPDATE. IT IS ENTERED FROM AND EXITS TO THE             *
*  A000-UPDATE-CUSTOMER-MASTER MODULE.                          *
*                                                               *
*****************************************************************

 B000-OBTAIN-UPDATED-DATA.

     MOVE 'NO ' TO HAS-UPDATED-DATA-BEEN-OBTAINED.
     PERFORM C000-OBTAIN-VALID-TRANS-DATA.
     PERFORM B001-SEARCH-FOR-VALID-DATA
          UNTIL UPDATED-DATA-HAS-BEEN-OBTAINED OR
                THERE-ARE-NO-MORE-TRAN-RECORDS.



 B001-SEARCH-FOR-VALID-DATA.

     MOVE SPACES TO WAS-IT-A-VALID-ADD-RECORD.
     MOVE SPACES TO WAS-IT-A-VALID-DELETE-RECORD.
     MOVE SPACES TO WAS-IT-A-VALID-CHANGE-RECORD.
     MOVE 'YES' TO WAS-A-MASTER-RECORD-FOUND.
     MOVE CUSTOMER-NUMBER-CURRENT-REC TO
          KEY-OF-THE-RECORD-TO-RETRIEVE.
     READ CUSTOMER-MASTER-I-O-FILE INTO MASTER-INPUT-WORK-AREA
          INVALID KEY
             MOVE 'NO ' TO WAS-A-MASTER-RECORD-FOUND.
     IF A-MASTER-RECORD-WAS-FOUND
        MOVE CUSTOMER-NUMBER-MSTRIN-WORK TO
             MASTER-RECORD-BEING-PROCESSED
     ELSE
        MOVE SPACES TO MASTER-RECORD-BEING-PROCESSED.
     IF (A-MASTER-RECORD-WAS-NOT-FOUND) OR
          (A-MASTER-RECORD-WAS-FOUND AND
          CUSTOMER-RECORD-IS-DELETED)
        PERFORM C020-PROCESS-UNEQUAL-RECORDS
        IF IT-WAS-A-VALID-ADD-RECORD
           MOVE 'YES' TO HAS-UPDATED-DATA-BEEN-OBTAINED
        ELSE
           PERFORM C000-OBTAIN-VALID-TRANS-DATA
     ELSE
        PERFORM C010-PROCESS-EQUAL-RECORDS
        IF IT-WAS-A-VALID-CHANGE-RECORD OR
              IT-WAS-A-VALID-DELETE-RECORD
           MOVE 'YES' TO HAS-UPDATED-DATA-BEEN-OBTAINED
        ELSE
           PERFORM C000-OBTAIN-VALID-TRANS-DATA.
*****************************************************************
*                                                               *
*  THE FUNCTION OF THIS MODULE IS TO WRITE THE UPDATED FILE.    *
*  IT IS ENTERED FROM AND EXITS TO THE A000-UPDATE-CUSTOMER-    *
*  MASTER MODULE.                                               *
*                                                               *
*****************************************************************

 B010-WRITE-UPDATED-FILE.

     MOVE SPACES TO DID-A-VALID-WRITE-OCCUR.
     IF MASTER-RECORD-BEING-PROCESSED =
          KEY-OF-THE-RECORD-TO-RETRIEVE
        PERFORM B011-REWRITE-THE-MASTER-RECORD
     ELSE
        PERFORM B012-WRITE-A-NEW-MASTER-RECORD.



 B011-REWRITE-THE-MASTER-RECORD.

     REWRITE CUSTOMER-MASTER-I-O-RECORD
          FROM MASTER-OUTPUT-WORK-AREA.



 B012-WRITE-A-NEW-MASTER-RECORD.

     MOVE CUSTOMER-NUMBER-CURRENT-REC TO
          KEY-OF-THE-RECORD-TO-RETRIEVE.
     WRITE CUSTOMER-MASTER-I-O-RECORD
          FROM MASTER-OUTPUT-WORK-AREA
          INVALID KEY
             MOVE 'NO ' TO DID-A-VALID-WRITE-OCCUR.
*****************************************************************
*                                                               *
*  THE FUNCTION OF THIS MODULE IS TO OBTAIN VALID TRANSACTION   *
*  DATA. IT IS ENTERED FROM AND EXITS TO THE                    *
*  B000-OBTAIN-UPDATED-DATA MODULE.                             *
*                                                               *
*****************************************************************

 C000-OBTAIN-VALID-TRANS-DATA.

     MOVE SPACES TO ARE-TRAN-RECORDS-VALID.
     PERFORM U000-OBTAIN-TRANS-RECORDS.
     PERFORM C001-READ-AND-EDIT
          UNTIL TRAN-RECORDS-ARE-VALID OR
                THERE-ARE-NO-MORE-TRAN-RECORDS.



 C001-READ-AND-EDIT.

     IF TRANSACTION-CODE-CURRENT-REC = TRANSACTION-TYPE-ONE
        PERFORM D010-EDIT-TYPE-1-RECORD
        PERFORM D020-EDIT-TYPE-2-RECORD
     ELSE IF TRANSACTION-CODE-CURRENT-REC = TRANSACTION-TYPE-THRE
        PERFORM D030-EDIT-TYPE-3-RECORD
     ELSE IF TRANSACTION-CODE-CURRENT-REC = TRANSACTION-TYPE-FOU
        PERFORM D040-EDIT-TYPE-4-RECORD
     ELSE
        MOVE CUSTOMER-NUMBER-CURRENT-REC TO CUSTOMER-NUMBER-WORK
        PERFORM U020-EDIT-CUSTOMER-NUMBER.
     IF TRAN-RECORDS-ARE-NOT-VALID
        MOVE SPACES TO ARE-TRAN-RECORDS-VALID
        PERFORM U000-OBTAIN-TRANS-RECORDS
     ELSE
        MOVE 'YES' TO ARE-TRAN-RECORDS-VALID.
*****************************************************************
*                                                               *
*  THE FUNCTION OF THIS MODULE IS TO PROCESS EQUAL MASTER AND   *
*  TRANSACTION RECORDS. IT IS ENTERED FROM AND EXITS TO THE     *
*  B000-OBTAIN-UPDATED-DATA MODULE.                             *
*                                                               *
*****************************************************************

 C010-PROCESS-EQUAL-RECORDS.

     IF TRANSACTION-CODE-CURRENT-REC = TRANSACTION-TYPE-THREE O
          TRANSACTION-CODE-CURRENT-REC = TRANSACTION-TYPE-FOUR
        PERFORM D050-PROCESS-CHANGE-TRANS
        MOVE MASTER-INPUT-WORK-AREA TO MASTER-OUTPUT-WORK-AREA
        MOVE 'YES' TO WAS-IT-A-VALID-CHANGE-RECORD
     ELSE
        IF TRANSACTION-CODE-CURRENT-REC = TRANSACTION-TYPE-FIVE
           MOVE DELETE-CODE-CONSTANT TO DELETE-CODE-MSTRIN-WORK
           MOVE MASTER-INPUT-WORK-AREA TO MASTER-OUTPUT-WORK-AREA
           MOVE 'YES' TO WAS-IT-A-VALID-DELETE-RECORD
           MOVE CUSTOMER-NUMBER-CURRENT-REC TO
                CUSTOMER-NUMBER-RPTWORK
           MOVE DELETION-MSG TO ACTION-TAKEN-RPTWORK
           PERFORM U000-WRITE-TRANSACTION-REPORT
        ELSE
           MOVE CUSTOMER-NUMBER-CURRENT-REC TO
                CUSTOMER-NUMBER-RPTWORK
           MOVE TRANSACTION-CODE-FIELD-MSG TO
                FIELD-CHANGED-ERROR-RPTWORK
           MOVE ADD-OF-DUPLICATE-ERROR-MSG TO
                TYPE-OF-ERROR-RPTWORK
           PERFORM U000-WRITE-TRANSACTION-REPORT
           MOVE 'NO ' TO WAS-IT-A-VALID-CHANGE-RECORD
           MOVE 'NO ' TO WAS-IT-A-VALID-DELETE-RECORD.
*****************************************************************
*                                                               *
*  THE FUNCTION OF THIS MODULE IS TO PROCESS AN UNEQUAL         *
*  TRANSACTION RECORD. IT IS ENTERED FROM AND EXITS TO THE      *
*  B000-OBTAIN-UPDATED-DATA MODULE.                             *
*                                                               *
*****************************************************************

 C020-PROCESS-UNEQUAL-RECORDS.

     IF TRANSACTION-CODE-CURRENT-REC = TRANSACTION-TYPE-ONE
        MOVE CUSTOMER-NUMBER-TRANSORT1 TO
             CUSTOMER-NUMBER-MSTROUT-WORK
        MOVE CUSTOMER-NAME-TRANSORT1 TO CUSTOMER-NAME-MSTROUT-WOR
        MOVE CUSTOMER-ADDRESS-TRANSORT1 TO
             CUSTOMER-ADDRESS-MSTROUT-WORK
        MOVE CUSTOMER-CITY-TRANSORT1 TO CUSTOMER-CITY-MSTROUT-WO
        MOVE CUSTOMER-STATE-TRANSORT1 TO
             CUSTOMER-STATE-MSTROUT-WORK
        MOVE CUSTOMER-ZIP-CODE-TRANSORT1 TO
             CUSTOMER-ZIP-CODE-MSTROUT-WORK
        MOVE CUSTOMER-DISCOUNT-TRANSORT2 TO
             CUSTOMER-DISCOUNT-MSTROUT-WORK
        MOVE LAST-ACTIVITY-DATE-TRANSORT2 TO
             LAST-ACTIVITY-MSTROUT-WORK
        MOVE SPACES TO DELETE-CODE-MSTROUT-WORK
        MOVE CUSTOMER-NUMBER-TRANSORT1 TO
             CUSTOMER-NUMBER-RPTWORK
        MOVE ADDITION-MSG TO ACTION-TAKEN-RPTWORK
        PERFORM U000-WRITE-TRANSACTION-REPORT
        MOVE 'YES' TO WAS-IT-A-VALID-ADD-RECORD
     ELSE
        MOVE CUSTOMER-NUMBER-CURRENT-REC TO
             CUSTOMER-NUMBER-RPTWORK
        MOVE TRANSACTION-CODE-FIELD-MSG TO
```

Source Listing (Part 4 of 7)

```
024160              FIELD-CHANGED-ERROR-RPTWORK
024170        MOVE NO-MASTER-RECORD-MSG TO TYPE-OF-ERROR-RPTWORK
024180        PERFORM U000-WRITE-TRANSACTION-REPORT
024190        MOVE 'NO ' TO WAS-IT-A-VALID-ADD-RECORD.
025010******************************************************************
025020*                                                                *
025030*  THIS MODULE OBTAINS A TRANSACTION RECORD(S) WITH VALID       *
025040*  TRANSACTION CODES. IT IS ENTERED FROM AND EXITS TO THE       *
025050*  C000-OBTAIN-VALID-TRANS-DATA MODULE.                          *
025060*                                                                *
025070******************************************************************
025080
025090 D000-OBTAIN-TRANS-RECORDS.
025100
025110     MOVE SPACES TO IS-TRANSACTION-CODE-VALID.
025120     IF TRANSACTION-FILE-IS-NOT-SORTED
025130        PERFORM E000-SORT-TRANSACTION-RECORDS
025140        OPEN INPUT SORTED-CUST-TRANS-INPUT-FILE
025150        MOVE 'YES' TO IS-TRANSACTION-FILE-SORTED.
025160     PERFORM D002-READ-FIRST-RECORD
025200     PERFORM U001-CHECK-FOR-VALID-CODES
026010           UNTIL TRANSACTION-CODE-IS-VALID OR
026020                 THERE-ARE-NO-MORE-TRAN-RECORDS.
026030     IF THERE-ARE-NO-MORE-TRAN-RECORDS
026040        CLOSE SORTED-CUST-TRANS-INPUT-FILE.
026050
026060
026070
026080 U001-CHECK-FOR-VALID-CODES.
026090
026100     IF TRANSACTION-CODE-TRANSORT1 = TRANSACTION-TYPE-THREE OR
026110                                   = TRANSACTION-TYPE-FOUR  OR
026120                                   = TRANSACTION-TYPE-FIVE
026130        MOVE TRANSACTION-CODE-TRANSORT1 TO
026140              TRANSACTION-CODE-CURRENT-REC
026150        MOVE CUSTOMER-NUMBER-TRANSORT1 TO
026160              CUSTOMER-NUMBER-CURRENT-REC
026170        MOVE 'YES' TO IS-TRANSACTION-CODE-VALID
026180        IF TRANSACTION-CODE-CURRENT-REC = TRANSACTION-TYPE-FOUR
026190           MOVE SORTED-TRANS-RECORD-TYPE-1 TO
026200                 SORTED-TRANS-RECORD-TYPE-2
026210        ELSE
026220           NEXT SENTENCE
026230     ELSE IF TRANSACTION-CODE-TRANSORT1 = TRANSACTION-TYPE-ONE
026240        PERFORM D003-READ-SECOND-RECORD
026250        IF A-TRAN-RECORD-WAS-READ
026260           IF TRANSACTION-CODE-TRANSORT2 = TRANSACTION-TYPE-TWO
026270                 AND CUSTOMER-NUMBER-TRANSORT1 =
026280                 CUSTOMER-NUMBER-TRANSORT2
026290              MOVE 'YES' TO IS-TRANSACTION-CODE-VALID
026300              MOVE TRANSACTION-CODE-TRANSORT1 TO
026310                    TRANSACTION-CODE-CURRENT-REC
026320              MOVE CUSTOMER-NUMBER-TRANSORT1 TO
026330                    CUSTOMER-NUMBER-CURRENT-REC
027010           ELSE
027020              MOVE CUSTOMER-NUMBER-TRANSORT1 TO
027030                    CUSTOMER-NUMBER-RPTWORK
027040              MOVE CUSTOMER-RECORDS-FIELD-MSG TO
027050                    FIELD-CHANGED-ERROR-RPTWORK
027060              MOVE TYPE1-BUT-NO-TYPE2-ERROR-MSG TO
027070                    TYPE-OF-ERROR-RPTWORK
027080              PERFORM U000-WRITE-TRANSACTION-REPORT
027090              MOVE SORTED-TRANS-RECORD-TYPE-2 TO
027100                    SORTED-TRANS-RECORD-TYPE-1
027110        ELSE
027120           MOVE CUSTOMER-NUMBER-TRANSORT1 TO
027130                 CUSTOMER-NUMBER-RPTWORK
027140           MOVE CUSTOMER-RECORDS-FIELD-MSG TO
027150                 FIELD-CHANGED-ERROR-RPTWORK
027160           MOVE TYPE1-BUT-NO-TYPE2-ERROR-MSG TO
027170                 TYPE-OF-ERROR-RPTWORK
027180           PERFORM U000-WRITE-TRANSACTION-REPORT
027190     ELSE IF TRANSACTION-CODE-TRANSORT1 = TRANSACTION-TYPE-TWO
027200        MOVE CUSTOMER-NUMBER-TRANSORT1 TO
028010              CUSTOMER-NUMBER-RPTWORK
028020        MOVE CUSTOMER-RECORDS-FIELD-MSG TO
028030              FIELD-CHANGED-ERROR-RPTWORK
028040        MOVE TYPE2-BUT-NO-TYPE1-ERROR-MSG TO
028050              TYPE-OF-ERROR-RPTWORK
028060        PERFORM U000-WRITE-TRANSACTION-REPORT
028070        PERFORM U002-READ-FIRST-RECORD
028080     ELSE
028090        MOVE CUSTOMER-NUMBER-TRANSORT1 TO
028100              CUSTOMER-NUMBER-RPTWORK
028110        MOVE TRANSACTION-CODE-FIELD-MSG TO
028120              FIELD-CHANGED-ERROR-RPTWORK
028130        MOVE BAD-TRANS-CODE-ERROR-MSG TO
028140              TYPE-OF-ERROR-RPTWORK
028150        PERFORM U000-WRITE-TRANSACTION-REPORT
028160        PERFORM D002-READ-FIRST-RECORD.
028170
028180
028190
028200 D002-READ-FIRST-RECORD.
028210
028220     READ SORTED-CUST-TRANS-INPUT-FILE
028230           INTO SORTED-TRANS-RECORD-TYPE-1
028240           AT END
028250              MOVE 'NO ' TO ARE-THERE-MORE-TRAN-RECORDS.
028260
028270
028280
028290 D003-READ-SECOND-RECORD.
028300
028310     READ SORTED-CUST-TRANS-INPUT-FILE
028320           INTO SORTED-TRANS-RECORD-TYPE-2
028330           AT END
028340              MOVE 'NO ' TO ARE-THERE-MORE-TRAN-RECORDS.
```

```
  028360******************************************************************
  028370*                                                                *
  028380*  THE FUNCTION OF THIS MODULE IS TO EDIT A TYPE 1 TRANSACTION. *
  029010*  IT IS ENTERED FROM AND EXITS TO THE C000-OBTAIN-VALID-TRANS-*
  029020*  DATA MODULE.                                                  *
  029030*                                                                *
  029040******************************************************************
  029050
  029060 D010-EDIT-TYPE-1-RECORD.
  029070
  029080     COPY EDTYP1
  029090            REPLACING CUSTOMER-NUMBER-EXCP-WORK BY
  029100                            CUSTOMER-NUMBER-RPTWORK
  029110                      FIELD-IN-ERROR-EXCP-WORK BY
  029120                            FIELD-CHANGED-ERROR-RPTWORK
  029130                      TYPE-OF-ERROR-EXCP-WORK BY
  029140                            TYPE-OF-ERROR-RPTWORK
  029150                      U000-WRITE-EXCEPTION-REPORT BY
  029160                            U000-WRITE-TRANSACTION-REPORT.
C 031180
C 031190     MOVE CUSTOMER-NUMBER-TRANSORT1 TO CUSTOMER-NUMBER-WORK.
C 031200     PERFORM U020-EDIT-CUSTOMER-NUMBER.
C 031210     IF CUSTOMER-NAME-TRANSORT1 = SPACES
C 032010        MOVE CUSTOMER-NUMBER-TRANSORT1 TO
C 032020              CUSTOMER-NUMBER-RPTWORK
C 032030        MOVE CUSTOMER-NAME-FIELD-MSG TO
C 032040              FIELD-CHANGED-ERROR-RPTWORK
C 032050        MOVE BLANK-FIELD-ERROR-MSG TO TYPE-OF-ERROR-RPTWORK
C 032060        PERFORM U000-WRITE-TRANSACTION-REPORT
C 032070        MOVE 'NO ' TO ARE-TRAN-RECORDS-VALID.
C 032080     IF CUSTOMER-ADDRESS-TRANSORT1 = SPACES
C 032090        MOVE CUSTOMER-NUMBER-TRANSORT1 TO
C 032100              CUSTOMER-NUMBER-RPTWORK
C 032110        MOVE CUSTOMER-ADDRESS-FIELD-MSG TO
C 032120              FIELD-CHANGED-ERROR-RPTWORK
C 032130        MOVE BLANK-FIELD-ERROR-MSG TO TYPE-OF-ERROR-RPTWORK
C 032140        PERFORM U000-WRITE-TRANSACTION-REPORT
C 032150        MOVE 'NO ' TO ARE-TRAN-RECORDS-VALID.
C 032160     IF CUSTOMER-CITY-TRANSORT1 = SPACES
C 032170        MOVE CUSTOMER-NUMBER-TRANSORT1 TO
C 032180              CUSTOMER-NUMBER-RPTWORK
C 032190        MOVE CUSTOMER-CITY-FIELD-MSG TO
C 032200              FIELD-CHANGED-ERROR-RPTWORK
C 033010        MOVE BLANK-FIELD-ERROR-MSG TO TYPE-OF-ERROR-RPTWORK
C 033020        PERFORM U000-WRITE-TRANSACTION-REPORT
C 033030        MOVE 'NO ' TO ARE-TRAN-RECORDS-VALID.
C 033040     SEARCH ALL VALID-STATE-TABLE
C 033050           AT END
C 033060              MOVE CUSTOMER-NUMBER-TRANSORT1 TO
C 033070                    CUSTOMER-NUMBER-RPTWORK
C 033080              MOVE CUSTOMER-STATE-FIELD-MSG TO
C 033090                    FIELD-CHANGED-ERROR-RPTWORK
C 033100              MOVE BAD-STATE-CODE-ERROR-MSG TO
C 033110                    TYPE-OF-ERROR-RPTWORK
C 033120              PERFORM U000-WRITE-TRANSACTION-REPORT
C 033130              MOVE 'NO ' TO ARE-TRAN-RECORDS-VALID
C 033140            WHEN STATE-CODE (STATE-CODE-INDEX) =
C 033150                  CUSTOMER-STATE-TRANSORT1
C 033160              NEXT SENTENCE.
C 033170     IF CUSTOMER-ZIP-CODE-TRANSORT1 IS NOT NUMERIC
C 033180        MOVE CUSTOMER-NUMBER-TRANSORT1 TO
C 033190              CUSTOMER-NUMBER-RPTWORK
C 033200        MOVE CUSTOMER-ZIP-CODE-FIELD-MSG TO
C 034010              FIELD-CHANGED-ERROR-RPTWORK
C 034020        MOVE FIELD-NOT-NUMERIC-ERROR-MSG TO
C 034030              TYPE-OF-ERROR-RPTWORK
C 034040        PERFORM U000-WRITE-TRANSACTION-REPORT
C 034050        MOVE 'NO ' TO ARE-TRAN-RECORDS-VALID.
  034070******************************************************************
  034080*                                                                *
  034090*  THE FUNCTION OF THIS MODULE IS TO EDIT A TYPE 2 TRANSACTION. *
  034100*  IT IS ENTERED FROM AND EXITS TO THE C000-OBTAIN-VALID-TRANS-*
  034110*  DATA MODULE.                                                  *
  034120*                                                                *
  034130******************************************************************
  034140
  034150 D020-EDIT-TYPE-2-RECORD.
  034160
  034170     COPY EDTYP2
  034180            REPLACING CUSTOMER-NUMBER-EXCP-WORK BY
  034190                            CUSTOMER-NUMBER-RPTWORK
  034200                      FIELD-IN-ERROR-EXCP-WORK BY
  035010                            FIELD-CHANGED-ERROR-RPTWORK
  035020                      TYPE-OF-ERROR-EXCP-WORK BY
  035030                            TYPE-OF-ERROR-RPTWORK
  035040                      U000-WRITE-EXCEPTION-REPORT BY
  035050                            U000-WRITE-TRANSACTION-REPORT.
C*034160
C 034170     MOVE CUSTOMER-NUMBER-TRANSORT2 TO CUSTOMER-NUMBER-WORK.
C 034180     PERFORM U020-EDIT-CUSTOMER-NUMBER.
C 034190     IF CUSTOMER-DISCOUNT-TRANSORT2 IS NOT NUMERIC
C 034200        MOVE CUSTOMER-NUMBER-TRANSORT2 TO
C 034210              CUSTOMER-NUMBER-RPTWORK
C 035010        MOVE CUSTOMER-DISCOUNT-FIELD-MSG TO
C 035020              FIELD-CHANGED-ERROR-RPTWORK
C 035030        MOVE FIELD-NOT-NUMERIC-ERROR-MSG TO
C 035040              TYPE-OF-ERROR-RPTWORK
C 035050        PERFORM U000-WRITE-TRANSACTION-REPORT
C 035060        MOVE 'NO ' TO ARE-TRAN-RECORDS-VALID
C 035070     ELSE
C 035080        IF CUSTOMER-DISCOUNT-TRANSORT2 IS GREATER THAN
C 035090              MAXIMUM-DISCOUNT-PERCENT
C 035100           MOVE CUSTOMER-NUMBER-TRANSORT2 TO
C 035110                 CUSTOMER-NUMBER-RPTWORK
C 035120           MOVE CUSTOMER-DISCOUNT-FIELD-MSG TO
C 035130                 FIELD-CHANGED-ERROR-RPTWORK
C 035140           MOVE BAD-DISCOUNT-ERROR-MSG TO
C 035150                 TYPE-OF-ERROR-RPTWORK
C 035160           PERFORM U000-WRITE-TRANSACTION-REPORT
C 035170           MOVE 'NO ' TO ARE-TRAN-RECORDS-VALID.
C 035180     IF LAST-ACTIVITY-DATE-TRANSORT2 IS NOT NUMERIC
C 035190        MOVE CUSTOMER-NUMBER-TRANSORT2 TO
C 035200              CUSTOMER-NUMBER-RPTWORK
C 036010        MOVE LAST-ACTIVITY-DATE-FIELD-MSG TO
```

BEST COPY

HO 3 (cont'd)

Source Listing (Part 5 of 7)

```
C 036020                FIELD-CHANGED-ERROR-RPTWORK
C 036030         MOVE FIELD-NOT-NUMERIC-ERROR-MSG TO
C 036040                TYPE-OF-ERROR-RPTWORK
C 036050         PERFORM U000-WRITE-TRANSACTION-REPORT
C 036060         MOVE 'NO ' TO ARE-TRAN-RECORDS-VALID
C 036070      ELSE
C 036080         IF NOT VALID-MONTH
C 036110            MOVE CUSTOMER-NUMBER-TRANSORT2 TO
C 036120                   CUSTOMER-NUMBER-RPTWORK
C 036130            MOVE LAST-ACTIVITY-DATE-FIELD-MSG TO
C 036140                   FIELD-CHANGED-ERROR-RPTWORK
C 036150            MOVE BAD-MONTH-ERROR-MSG TO TYPE-OF-ERROR-RPTWORK
C 036160            PERFORM U000-WRITE-TRANSACTION-REPORT
C 036170            MOVE 'NO ' TO ARE-TRAN-RECORDS-VALID
C 036180         ELSE
C 036190            SET MONTH-DAY-INDEX TO MONTH-LAST-ACTIVITY-TRANSORT2
C 036200            IF DAY-LAST-ACTIVITY-TRANSORT2 IS LESS THAN
C 037010                   FIRST-DAY-OF-MONTH-NUMBER OR
C 037020                   DAY-LAST-ACTIVITY-TRANSORT2 IS GREATER THAN
C 037030                   NUMBER-OF-DAYS-IN-MONTH (MONTH-DAY-INDEX)
C 037040               MOVE CUSTOMER-NUMBER-TRANSORT2 TO
C 037050                      CUSTOMER-NUMBER-RPTWORK
C 037060               MOVE LAST-ACTIVITY-DATE-FIELD-MSG TO
C 037070                      FIELD-CHANGED-ERROR-RPTWORK
C 037080               MOVE BAD-DAY-ERROR-MSG TO TYPE-OF-ERROR-RPTWORK
C 037090               PERFORM U000-WRITE-TRANSACTION-REPORT
C 037100               MOVE 'NO ' TO ARE-TRAN-RECORDS-VALID.
  035070**************************************************************
  035080*                                                             *
  035090*  THIS MODULE EDITS A TYPE 3 TRANSACTION RECORD. IT IS       *
  035100*  ENTERED FROM AND EXITS TO THE C000-OBTAIN-VALID-TRANS-DATA *
  035110*  MODULE.                                                    *
  035120*                                                             *
  035130**************************************************************
  035140
  035150 D030-EDIT-TYPE-3-RECORD.
  035160
  035170     MOVE CUSTOMER-NUMBER-TRANSORT1 TO CUSTOMER-NUMBER-WORK.
  035180     PERFORM U020-EDIT-CUSTOMER-NUMBER.
  035190     IF CUSTOMER-STATE-TRANSORT1 NOT = SPACES
  035200        MOVE CUSTOMER-NUMBER-TRANSORT1 TO
  035210               CUSTOMER-NUMBER-STATEDIT-WORK
  035220        MOVE CUSTOMER-STATE-TRANSORT1 TO
  035230               STATE-EDIT-WORK
  035240        PERFORM E010-EDIT-STATE-CODE
  036010        IF CUSTOMER-ZIP-CODE-TRANSORT1 = SPACES
  036020           MOVE CUSTOMER-NUMBER-TRANSORT1 TO
  036030                  CUSTOMER-NUMBER-RPTWORK
  036040           MOVE CUSTOMER-ZIP-CODE-FIELD-MSG TO
  036050                  FIELD-CHANGED-ERROR-RPTWORK
  036060           MOVE STATE-BUT-NO-ZIP-ERROR-MSG TO
  036070                  TYPE-OF-ERROR-RPTWORK
  036080           PERFORM U000-WRITE-TRANSACTION-REPORT
  036090           MOVE 'NO ' TO ARE-TRAN-RECORDS-VALID.
  036100     IF CUSTOMER-ZIP-CODE-TRANSORT1 NOT = SPACES AND
  036110           CUSTOMER-ZIP-CODE-TRANSORT1 IS NOT NUMERIC
  036120        MOVE CUSTOMER-NUMBER-TRANSORT1 TO
  036130               CUSTOMER-NUMBER-RPTWORK
  036140        MOVE CUSTOMER-ZIP-CODE-FIELD-MSG TO
  036150               FIELD-CHANGED-ERROR-RPTWORK
  036160        MOVE FIELD-NOT-NUMERIC-ERROR-MSG TO
  036170               TYPE-OF-ERROR-RPTWORK
  036180        PERFORM U000-WRITE-TRANSACTION-REPORT
  036190        MOVE 'NO ' TO ARE-TRAN-RECORDS-VALID.
  037010**************************************************************
  037020*                                                             *
  037030*  THIS MODULE EDITS A TYPE 4 TRANSACTION RECORD. IT IS ENTERED
  037040*  FROM AND EXITS TO THE C000-OBTAIN-VALID-TRANS-DATA MODULE.
  037050*
  037060**************************************************************
  037070
  037080 D040-EDIT-TYPE-4-RECORD.
  037090
  037100     MOVE CUSTOMER-NUMBER-TRANSORT2 TO CUSTOMER-NUMBER-WORK.
  037110     PERFORM U020-EDIT-CUSTOMER-NUMBER.
  037120     IF CUSTOMER-DISCOUNT-TRANSORT2 NOT = SPACES
  037130        MOVE CUSTOMER-NUMBER-TRANSORT2 TO
  037140               CUSTOMER-NUMBER-DISCEDIT-WORK
  037150        MOVE CUSTOMER-DISCOUNT-TRANSORT2 TO
  037160               DISCOUNT-EDIT-WORK
  037170        PERFORM E020-EDIT-DISCOUNT.
  037180     IF LAST-ACTIVITY-DATE-TRANSORT2 NOT = SPACES
  037190        MOVE CUSTOMER-NUMBER-TRANSORT2 TO
  037200               CUSTOMER-NUMBER-DATEDIT-WORK
  038010        MOVE LAST-ACTIVITY-DATE-TRANSORT2 TO
  038020               LAST-ACTIVITY-EDIT-WORK
  038030        PERFORM E030-EDIT-LAST-ACTIVITY-DATE.
  038050**************************************************************
  038060*                                                             *
  038070*  THIS MODULE PROCESSES CHANGE TRANSACTIONS. IT IS ENTERED   *
  038080*  FROM AND EXITS TO THE C010-PROCESS-EQUAL-RECORDS MODULE.   *
  038090*                                                             *
  038100**************************************************************
  038110
  038120 D050-PROCESS-CHANGE-TRANS.
  038130
  038140     IF TRANSACTION-CODE-CURRENT-REC = TRANSACTION-TYPE-THREE
  038150        PERFORM D051-CHECK-TYPE-THREE-TRANS
  038160     ELSE
  038170        PERFORM D052-CHECK-TYPE-FOUR-TRANS.
  038180
  038190
  038200
  039010 D051-CHECK-TYPE-THREE-TRANS.
  039020
  039030     MOVE CUSTOMER-NUMBER-CURRENT-REC TO
  039040            CUSTOMER-NUMBER-TYPE-3-CHANGE.
  039050     IF CUSTOMER-ADDRESS-TRANSORT1 NOT = SPACES
  039060        MOVE CUSTOMER-ADDRESS-TRANSORT1 TO ADDRESS-CHANGE-WORK
  039070        PERFORM E040-CHANGE-CUSTOMER-ADDRESS.
  039080     IF CUSTOMER-CITY-TRANSORT1 NOT = SPACES
  039090        MOVE CUSTOMER-CITY-TRANSORT1 TO CITY-CHANGE-WORK
  039100        PERFORM E050-CHANGE-CUSTOMER-CITY.
```

```
  039110     IF CUSTOMER-STATE-TRANSORT1 NOT = SPACES
  039120        MOVE CUSTOMER-STATE-TRANSORT1 TO STATE-CHANGE-WORK
  039130        PERFORM E060-CHANGE-CUSTOMER-STATE.
  039140     IF CUSTOMER-ZIP-CODE-TRANSORT1 NOT = SPACES
  039150        MOVE CUSTOMER-ZIP-CODE-TRANSORT1 TO ZIP-CODE-CHANGE-WORK
  039160        PERFORM E070-CHANGE-CUSTOMER-ZIP-CODE.
  039170
  039180
  039190
  039200 D052-CHECK-TYPE-FOUR-TRANS.
  040010
  040020     MOVE CUSTOMER-NUMBER-CURRENT-REC TO
  040030            CUSTOMER-NUMBER-TYPE-4-CHANGE.
  040040     IF CUSTOMER-DISCOUNT-TRANSORT2 NOT = SPACES
  040050        MOVE CUSTOMER-DISCOUNT-TRANSORT2 TO DISCOUNT-CHANGE-WORK
  040060        PERFORM E080-CHANGE-CUSTOMER-DISCOUNT.
  040070     IF LAST-ACTIVITY-DATE-TRANSORT2 NOT = SPACES
  040080        MOVE LAST-ACTIVITY-DATE-TRANSORT2 TO
  040090               LAST-ACTIVITY-CHANGE-WORK
  040100        PERFORM E090-CHANGE-LAST-ACTIVITY-DATE.
  040120**************************************************************
  040130*                                                             *
  040140*  THIS MODULE SORTS THE TRANSACTION INPUT FILE INTO          *
  040150*  TRANSACTION CODE WITHIN CUSTOMER NUMBER SEQUENCE. IT IS    *
  040160*  ENTERED FROM AND EXITS TO THE D000-OBTAIN-TRANS-RECORDS    *
  040170*  MODULE.                                                    *
  040180*                                                             *
  040190**************************************************************
  040200
  040210 E000-SORT-TRANSACTION-RECORDS.
  041010
  041020     SORT TRANSACTION-SORT-WORK-FILE
  041030            ON ASCENDING KEY CUSTOMER-NUMBER-SORTREC
  041040                             TRANSACTION-CODE-SORTREC
  041050            USING  CUSTOMER-TRANS-INPUT-FILE
  041060            GIVING SORTED-CUST-TRANS-INPUT-FILE.
  041080**************************************************************
  041090*                                                             *
  041100*  THIS MODULE IS ENTERED TO EDIT THE STATE CODE. IT IS       *
  041110*  ENTERED FROM AND EXITS TO THE D030-EDIT-TYPE-3-RECORD      *
  041120*  MODULE.                                                    *
  041130*                                                             *
  041140**************************************************************
  041150
  041160 E010-EDIT-STATE-CODE.
  041170
  041180     COPY EDSTCODE
  041190            REPLACING CUSTOMER-NUMBER-TRANSORT1 BY
  041200                         CUSTOMER-NUMBER-STATEDIT-WORK
  042010                      CUSTOMER-NUMBER-EXCP-WORK BY
  042020                         CUSTOMER-NUMBER-RPTWORK
  042030                      FIELD-IN-ERROR-EXCP-WORK BY
  042040                         FIELD-CHANGED-ERROR-RPTWORK
  042050                      TYPE-OF-ERROR-EXCP-WORK BY
  042060                         TYPE-OF-ERROR-RPTWORK
  042070                      U000-WRITE-EXCEPTION-REPORT BY
  042080                         U000-WRITE-TRANSACTION-REPORT
  042090                      CUSTOMER-STATE-TRANSORT1 BY
  042100                         STATE-EDIT-WORK.
C*033040     SEARCH ALL VALID-STATE-TABLE
C 033050            AT END
C 033060               MOVE CUSTOMER-NUMBER-STATEDIT-WORK TO
C 033070                      CUSTOMER-NUMBER-RPTWORK
C 033080               MOVE CUSTOMER-STATE-FIELD-MSG TO
C 033090                      FIELD-CHANGED-ERROR-RPTWORK
C 033100               MOVE BAD-STATE-CODE-ERROR-MSG TO
C 033110                      TYPE-OF-ERROR-RPTWORK
C 033120               PERFORM U000-WRITE-TRANSACTION-REPORT
C 033130               MOVE 'NO ' TO ARE-TRAN-RECORDS-VALID
C 033140            WHEN STATE-CODE (STATE-CODE-INDEX) =
C 033150                  STATE-EDIT-WORK
C 033160               NEXT SENTENCE.
  042120**************************************************************
  042130*                                                             *
  042140*  THIS MODULE EDITS THE CUSTOMER DISCOUNT. IT IS ENTERED FROM *
  042150*  AND EXITS TO THE D040-EDIT-TYPE-4-RECORD MODULE.           *
  042160*                                                             *
  042170**************************************************************
  042180
  042190 E020-EDIT-DISCOUNT.
  042200
  043010     COPY EDDISC
  043020            REPLACING CUSTOMER-DISCOUNT-TRANSORT2 BY
  043030                         DISCOUNT-EDIT-WORK
  043040                      CUSTOMER-NUMBER-TRANSORT2 BY
  043050                         CUSTOMER-NUMBER-DISCEDIT-WORK
  043060                      CUSTOMER-NUMBER-EXCP-WORK BY
  043070                         CUSTOMER-NUMBER-RPTWORK
  043080                      FIELD-IN-ERROR-EXCP-WORK BY
  043090                         FIELD-CHANGED-ERROR-RPTWORK
  043100                      TYPE-OF-ERROR-EXCP-WORK BY
  043110                         TYPE-OF-ERROR-RPTWORK
  043120                      U000-WRITE-EXCEPTION-REPORT BY
  043130                         U000-WRITE-TRANSACTION-REPORT.
C*034190     IF DISCOUNT-EDIT-WORK IS NOT NUMERIC
C 034200        MOVE CUSTOMER-NUMBER-DISCEDIT-WORK TO
C 034210               CUSTOMER-NUMBER-RPTWORK
C 035010        MOVE CUSTOMER-DISCOUNT-FIELD-MSG TO
C 035020               FIELD-CHANGED-ERROR-RPTWORK
C 035030        MOVE FIELD-NOT-NUMERIC-ERROR-MSG TO
C 035040               TYPE-OF-ERROR-RPTWORK
C 035050        PERFORM U000-WRITE-TRANSACTION-REPORT
C 035060        MOVE 'NO ' TO ARE-TRAN-RECORDS-VALID
C 035070     ELSE
C 035080        IF DISCOUNT-EDIT-WORK IS GREATER THAN
C 035090               MAXIMUM-DISCOUNT-PERCENT
C 035100           MOVE CUSTOMER-NUMBER-DISCEDIT-WORK TO
C 035110                  CUSTOMER-NUMBER-RPTWORK
C 035120           MOVE CUSTOMER-DISCOUNT-FIELD-MSG TO
C 035130                  FIELD-CHANGED-ERROR-RPTWORK
C 035140           MOVE BAD-DISCOUNT-ERROR-MSG TO
C 035150                  TYPE-OF-ERROR-RPTWORK
C 035160           PERFORM U000-WRITE-TRANSACTION-REPORT
C 035170           MOVE 'NO ' TO ARE-TRAN-RECORDS-VALID.
```

BEST COPY

Source Listing (Part 6 of 7)

```
*****************************************************************
*                                                               *
*  THIS MODULE EDITS THE LAST ACTIVITY DATE. IT IS ENTERED FROM *
*  AND EXITS TO THE D040-EDIT-TYPE-4 RECORD MODULE.             *
*                                                               *
*****************************************************************

 E030-EDIT-LAST-ACTIVITY-DATE.

     COPY EDACDATE
          REPLACING LAST-ACTIVITY-DATE-TRANSORT2 BY
                            LAST-ACTIVITY-EDIT-WORK
                    CUSTOMER-NUMBER-TRANSORT2 BY
                            CUSTOMER-NUMBER-DATEDIT-WORK
                    VALID-MONTH BY VALID-MONTH-WORK
                    MONTH-LAST-ACTIVITY-TRANSORT2 BY
                            MONTH-LAST-ACTIVITY-WORK
                    DAY-LAST-ACTIVITY-TRANSORT2 BY
                            DAY-LAST-ACTIVITY-WORK
                    CUSTOMER-NUMBER-EXCP-WORK BY
                            CUSTOMER-NUMBER-RPTWORK
                    FIELD-IN-ERROR-EXCP-WORK BY
                            FIELD-CHANGED-ERROR-RPTWORK
                    TYPE-OF-ERROR-EXCP-WORK BY
                            TYPE-OF-ERROR-RPTWORK
                    U000-WRITE-EXCEPTION-REPORT BY
                            U000-WRITE-TRANSACTION-REPORT.
     IF LAST-ACTIVITY-EDIT-WORK IS NOT NUMERIC
         MOVE CUSTOMER-NUMBER-DATEDIT-WORK TO
                 CUSTOMER-NUMBER-RPTWORK
         MOVE LAST-ACTIVITY-DATE-FIELD-MSG TO
                 FIELD-CHANGED-ERROR-RPTWORK
         MOVE FIELD-NOT-NUMERIC-ERROR-MSG TO
                 TYPE-OF-ERROR-RPTWORK
         PERFORM U000-WRITE-TRANSACTION-REPORT
         MOVE 'NO ' TO ARE-TRAN-RECORDS-VALID
     ELSE
         IF NOT VALID-MONTH-WORK
             MOVE CUSTOMER-NUMBER-DATEDIT-WORK TO
                     CUSTOMER-NUMBER-RPTWORK
             MOVE LAST-ACTIVITY-DATE-FIELD-MSG TO
                     FIELD-CHANGED-ERROR-RPTWORK
             MOVE BAD-MONTH-ERROR-MSG TO TYPE-OF-ERROR-RPTWORK
             PERFORM U000-WRITE-TRANSACTION-REPORT
             MOVE 'NO ' TO ARE-TRAN-RECORDS-VALID
         ELSE
             SET MONTH-DAY-INDEX TO MONTH-LAST-ACTIVITY-WORK
             IF DAY-LAST-ACTIVITY-WORK IS LESS THAN
                     FIRST-DAY-OF-MONTH-NUMBER OR
                     DAY-LAST-ACTIVITY-WORK IS GREATER THAN
                     NUMBER-OF-DAYS-IN-MONTH (MONTH-DAY-INDEX)
                 MOVE CUSTOMER-NUMBER-DATEDIT-WORK TO
                         CUSTOMER-NUMBER-RPTWORK
                 MOVE LAST-ACTIVITY-DATE-FIELD-MSG TO
                         FIELD-CHANGED-ERROR-RPTWORK
                 MOVE BAD-DAY-ERROR-MSG TO TYPE-OF-ERROR-RPTWORK
                 PERFORM U000-WRITE-TRANSACTION-REPORT
                 MOVE 'NO ' TO ARE-TRAN-RECORDS-VALID.
*****************************************************************
*                                                               *
*  THIS MODULE CHANGES THE CUSTOMER ADDRESS. IT IS ENTERED FROM *
*  AND EXITS TO THE D050-PROCESS-CHANGE-TRANS MODULE.           *
*                                                               *
*****************************************************************

 E040-CHANGE-CUSTOMER-ADDRESS.

     MOVE CUSTOMER-ADDRESS-MSTRIN-WORK TO
             OLD-MASTER-RECORD-RPTWORK.
     MOVE ADDRESS-CHANGE-WORK TO
             NEW-MASTER-RECORD-RPTWORK.
     MOVE CUSTOMER-NUMBER-TYPE-3-CHANGE TO
             CUSTOMER-NUMBER-RPTWORK.
     MOVE CUSTOMER-ADDRESS-FIELD-MSG TO
             FIELD-CHANGED-ERROR-RPTWORK.
     MOVE CHANGE-MSG TO ACTION-TAKEN-RPTWORK.
     PERFORM U000-WRITE-TRANSACTION-REPORT.
     MOVE ADDRESS-CHANGE-WORK TO
             CUSTOMER-ADDRESS-MSTRIN-WORK.
*****************************************************************
*                                                               *
*  THIS MODULE CHANGES THE CUSTOMER CITY. IT IS ENTERED FROM    *
*  AND EXITS TO THE D050-PROCESS-CHANGE-TRANS MODULE.           *
*                                                               *
*****************************************************************

 E050-CHANGE-CUSTOMER-CITY.

     MOVE CUSTOMER-CITY-MSTRIN-WORK TO
             OLD-MASTER-RECORD-RPTWORK.
     MOVE CITY-CHANGE-WORK TO
             NEW-MASTER-RECORD-RPTWORK.
     MOVE CUSTOMER-NUMBER-TYPE-3-CHANGE TO
             CUSTOMER-NUMBER-RPTWORK.
     MOVE CUSTOMER-CITY-FIELD-MSG TO
             FIELD-CHANGED-ERROR-RPTWORK.
     MOVE CHANGE-MSG TO ACTION-TAKEN-RPTWORK.
     PERFORM U000-WRITE-TRANSACTION-REPORT.
     MOVE CITY-CHANGE-WORK TO
             CUSTOMER-CITY-MSTRIN-WORK.
*****************************************************************
*                                                               *
*  THIS MODULE CHANGES THE CUSTOMER STATE. IT IS ENTERED FROM   *
*  AND EXITS TO THE D050-PROCESS-CHANGE-TRANS MODULE.           *
*                                                               *
*****************************************************************

 E060-CHANGE-CUSTOMER-STATE.

     MOVE CUSTOMER-STATE-MSTRIN-WORK TO
             OLD-MASTER-RECORD-RPTWORK.
     MOVE STATE-CHANGE-WORK TO
             NEW-MASTER-RECORD-RPTWORK.
     MOVE CUSTOMER-NUMBER-TYPE-3-CHANGE TO
             CUSTOMER-NUMBER-RPTWORK.
     MOVE CUSTOMER-STATE-FIELD-MSG TO
             FIELD-CHANGED-ERROR-RPTWORK.
     MOVE CHANGE-MSG TO ACTION-TAKEN-RPTWORK.
     PERFORM U000-WRITE-TRANSACTION-REPORT.
     MOVE STATE-CHANGE-WORK TO
             CUSTOMER-STATE-MSTRIN-WORK.
*****************************************************************
*                                                               *
*  THIS MODULE CHANGES THE CUSTOMER ZIP CODE. IT IS ENTERED     *
*  FROM AND EXITS TO THE D050-PROCESS-CHANGE-TRANS MODULE.      *
*                                                               *
*****************************************************************

 E070-CHANGE-CUSTOMER-ZIP-CODE.

     MOVE CUSTOMER-ZIP-CODE-MSTRIN-WORK TO
             OLD-MASTER-RECORD-RPTWORK.
     MOVE ZIP-CODE-CHANGE-WORK TO
             NEW-MASTER-RECORD-RPTWORK.
     MOVE CUSTOMER-NUMBER-TYPE-3-CHANGE TO
             CUSTOMER-NUMBER-RPTWORK.
     MOVE CUSTOMER-ZIP-CODE-FIELD-MSG TO
             FIELD-CHANGED-ERROR-RPTWORK.
     MOVE CHANGE-MSG TO ACTION-TAKEN-RPTWORK.
     PERFORM U000-WRITE-TRANSACTION-REPORT.
     MOVE ZIP-CODE-CHANGE-WORK TO
             CUSTOMER-ZIP-CODE-MSTRIN-WORK.
*****************************************************************
*                                                               *
*  THIS MODULE CHANGES THE CUSTOMER DISCOUNT. IT IS ENTERED     *
*  FROM AND EXITS TO THE D050-PROCESS-CHANGE-TRANS MODULE.      *
*                                                               *
*****************************************************************

 E080-CHANGE-CUSTOMER-DISCOUNT.

     MOVE CUSTOMER-DISCOUNT-MSTRIN-WORK TO
             OLD-MASTER-RECORD-RPTWORK.
     MOVE DISCOUNT-CHANGE-WORK TO
             NEW-MASTER-RECORD-RPTWORK.
     MOVE CUSTOMER-NUMBER-TYPE-4-CHANGE TO
             CUSTOMER-NUMBER-RPTWORK.
     MOVE CUSTOMER-DISCOUNT-FIELD-MSG TO
             FIELD-CHANGED-ERROR-RPTWORK.
     MOVE CHANGE-MSG TO ACTION-TAKEN-RPTWORK.
     PERFORM U000-WRITE-TRANSACTION-REPORT.
     MOVE DISCOUNT-CHANGE-WORK TO
             CUSTOMER-DISCOUNT-MSTRIN-WORK.
*****************************************************************
*                                                               *
*  THIS MODULE CHANGES THE LAST ACTIVITY DATE. IT IS ENTERED    *
*  FROM AND EXITS TO THE D050-PROCESS-CHANGE-TRANS MODULE.      *
*                                                               *
*****************************************************************

 E090-CHANGE-LAST-ACTIVITY-DATE.

     MOVE LAST-ACTIVITY-MSTRIN-WORK TO
             OLD-MASTER-RECORD-RPTWORK.
     MOVE LAST-ACTIVITY-CHANGE-WORK TO
             NEW-MASTER-RECORD-RPTWORK.
     MOVE CUSTOMER-NUMBER-TYPE-4-CHANGE TO
             CUSTOMER-NUMBER-RPTWORK.
     MOVE LAST-ACTIVITY-DATE-FIELD-MSG TO
             FIELD-CHANGED-ERROR-RPTWORK.
     MOVE CHANGE-MSG TO ACTION-TAKEN-RPTWORK.
     PERFORM U000-WRITE-TRANSACTION-REPORT.
     MOVE LAST-ACTIVITY-CHANGE-WORK TO
             LAST-ACTIVITY-MSTRIN-WORK.
*****************************************************************
*                                                               *
*  THE FUNCTION OF THIS MODULE IS TO WRITE THE TRANSACTION      *
*  REPORT. IT IS ENTERED FROM AND EXITS TO VARIOUS MODULES IN   *
*  THE PROGRAM (SEE HIERARCHY CHART).                           *
*                                                               *
*****************************************************************

 U000-WRITE-TRANSACTION-REPORT.

     IF THIS-IS-THE-FIRST-PAGE OR
             NUMBER-OF-LINES-PRINTED = PAGE-SIZE OR
             IS GREATER THAN PAGE-SIZE
         PERFORM U010-PRINT-HEADINGS
         MOVE SPACES TO UPDATE-TRANSACTION-REPORT-LINE
         MOVE CUSTOMER-NUMBER-RPTWORK TO CUSTOMER-NUMBER-REPORT.
     IF CUSTOMER-NUMBER-RPTWORK NOT = PREVIOUS-CUSTOMER-NUMBER
         MOVE CUSTOMER-NUMBER-RPTWORK TO CUSTOMER-NUMBER-REPORT
         MOVE CUSTOMER-NUMBER-RPTWORK TO PREVIOUS-CUSTOMER-NUMBER
         MOVE SPACE-TWO-LINES TO PROPER-SPACING.
     MOVE ACTION-TAKEN-RPTWORK TO ACTION-TAKEN-REPORT.
     MOVE FIELD-CHANGED-ERROR-RPTWORK TO
             FIELD-CHANGED-ERROR-REPORT.
     MOVE TYPE-OF-ERROR-RPTWORK TO TYPE-OF-ERROR-REPORT.
     MOVE OLD-MASTER-RECORD-RPTWORK TO OLD-MASTER-RECORD-REPORT.
     MOVE NEW-MASTER-RECORD-RPTWORK TO NEW-MASTER-RECORD-REPORT.
     WRITE UPDATE-TRANSACTION-REPORT-LINE
             AFTER PROPER-SPACING.
     ADD PROPER-SPACING TO NUMBER-OF-LINES-PRINTED.
     MOVE SPACE-ONE-LINE TO PROPER-SPACING.
     MOVE SPACES TO UPDATE-TRANSACTION-REPORT-LINE.
     MOVE SPACES TO TRANSACTION-REPORT-WORK-AREA.
*****************************************************************
*                                                               *
*  THIS MODULE PRINTS THE HEADINGS ON THE TRANSACTION REPORT.   *
*  IT IS ENTERED FROM AND EXITS TO THE U000-WRITE-TRANSACTION-  *
*  REPORT MODULE.                                               *
*                                                               *
*****************************************************************

 U010-PRINT-HEADINGS.
```

BEST COPY

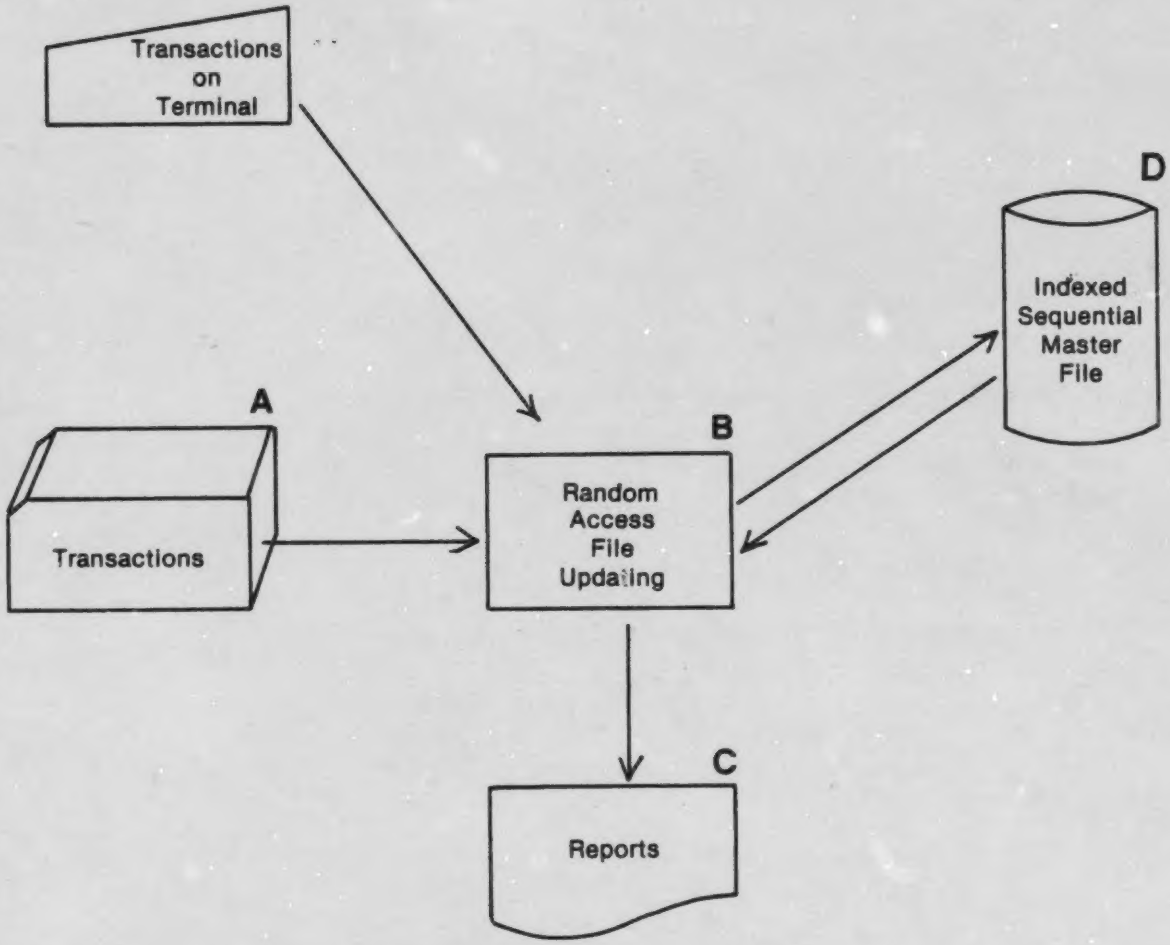
Source Listing (Part 7 of 7)

```
053120
053130     MOVE CURRENT-DATE TO DATE-HDG1.
053140     MOVE PAGE-NUMBER-COUNT TO PAGE-NUMBER-HDG1.
053150     WRITE UPDATE-TRANSACTION-REPORT-LINE FROM FIRST-HEADING-LINE
053160             AFTER ADVANCING TO-THE-TOP-OF-THE-PAGE.
053170     ADD 1 TO PAGE-NUMBER-COUNT.
053180     WRITE UPDATE-TRANSACTION-REPORT-LINE FROM SECOND-HEADING-LINE
053190             AFTER ADVANCING 2 LINES.
053200     WRITE UPDATE-TRANSACTION-REPORT-LINE FROM THIRD-HEADING-LINE
054010             AFTER ADVANCING 1 LINES.
054020     MOVE SPACE-TWO-LINES TO PROPER-SPACING.
054030     MOVE ZERO TO NUMBER-OF-LINES-PRINTED.
054050*****************************************************************
054060*                                                               *
054070*  THIS MODULE EDITS THE CUSTOMER NUMBER. IT IS ENTERED FROM    *
054080*  AND EXITS TO VARIOUS MODULES WITHIN THE PROGRAM (SEE         *
054090*  HIERARCHY CHART).                                            *
054100*                                                               *
054110*****************************************************************
054120
054130 U020-EDIT-CUSTOMER-NUMBER.
054140
054150     COPY EDCUSTNC
054160             REPLACING CUSTOMER-NUMBER-EXCP-WORK BY
054170                               CUSTOMER-NUMBER-RPTWORK
054180                       FIELD-IN-ERROR-EXCP-WORK BY
054190                               FIELD-CHANGED-ERROR-RPTWORK
054200                       TYPE-OF-ERROR-EXCP-WORK BY
055010                               TYPE-OF-ERROR-RPTWORK
055020                       U000-WRITE-EXCEPTION-REPORT BY
055030                               U000-WRITE-TRANSACTION-REPORT.
C*044040
C 044050     IF (FIRST-FOUR-CHAR-CUST-NUMB-WORK IS NOT NUMERIC) OR
C 044060             (FIFTH-CHAR-CUST-NUMB-WORK IS NOT NUMERIC AND
C 044070              NOT VALID-CHECK-DIGIT-FOR-TEN)
C 044080         MOVE CUSTOMER-NUMBER-WORK TO CUSTOMER-NUMBER-RPTWORK
C 044090         MOVE CUSTOMER-NUMBER-FIELD-MSG TO
C 044100                 FIELD-CHANGED-ERROR-RPTWORK
C 044110         MOVE FIELD-NOT-NUMERIC-ERROR-MSG TO
C 044120                 TYPE-OF-ERROR-RPTWORK
C 044130         PERFORM U000-WRITE-TRANSACTION-REPORT
C 044140         MOVE 'NO ' TO ARE-TRAN-RECORDS-VALID
C 044150     ELSE
C 044160         MOVE CUSTOMER-NUMBER-WORK TO
C 044170                 FIELD-TO-BE-CHECKED-MOD11-WORK
C 044180         CALL 'MOD11CHK' USING MOD-11-PASS-AREA-WORK
C 044190         IF CUST-NUMBER-DID-NOT-PASS-MOD11
C 044200             MOVE CUSTOMER-NUMBER-WORK TO CUSTOMER-NUMBER-RPTWORK
C 044210             MOVE CUSTOMER-NUMBER-FIELD-MSG TO
C 044220                     FIELD-CHANGED-ERROR-RPTWORK
C 045010             MOVE FAILS-MOD11-ERROR-MSG TO
C 045020                     TYPE-OF-ERROR-RPTWORK
C 045030             PERFORM U000-WRITE-TRANSACTION-REPORT
C 045040             MOVE 'NO ' TO ARE-TRAN-RECORDS-VALID.
```

BEST COPY

TM 1

Overview of Random Update


Transactions on Terminal
D
Indexed Sequential Master File
A
B
Transactions
Random Access File Updating
C
Reports

TM 2

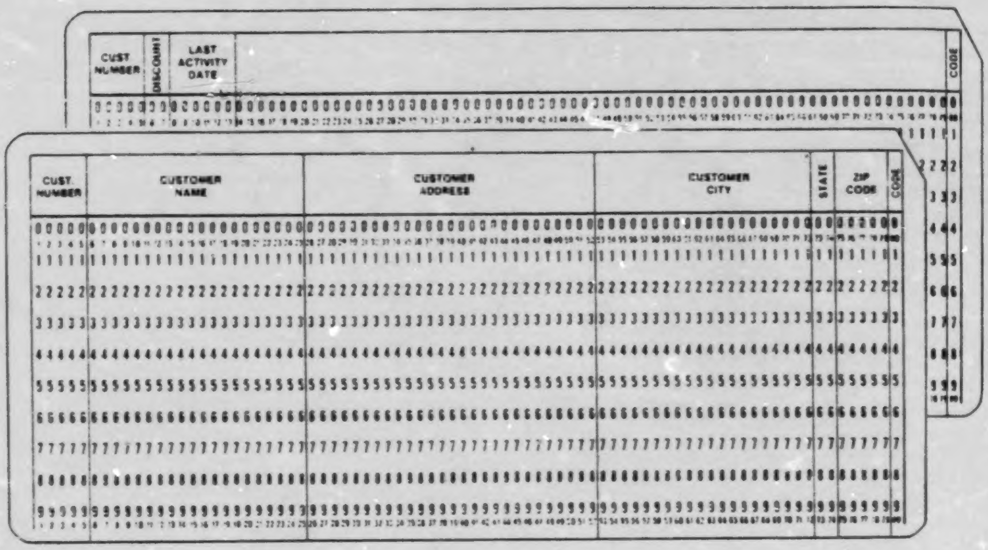
Format of Customer Master File

Customer Master File

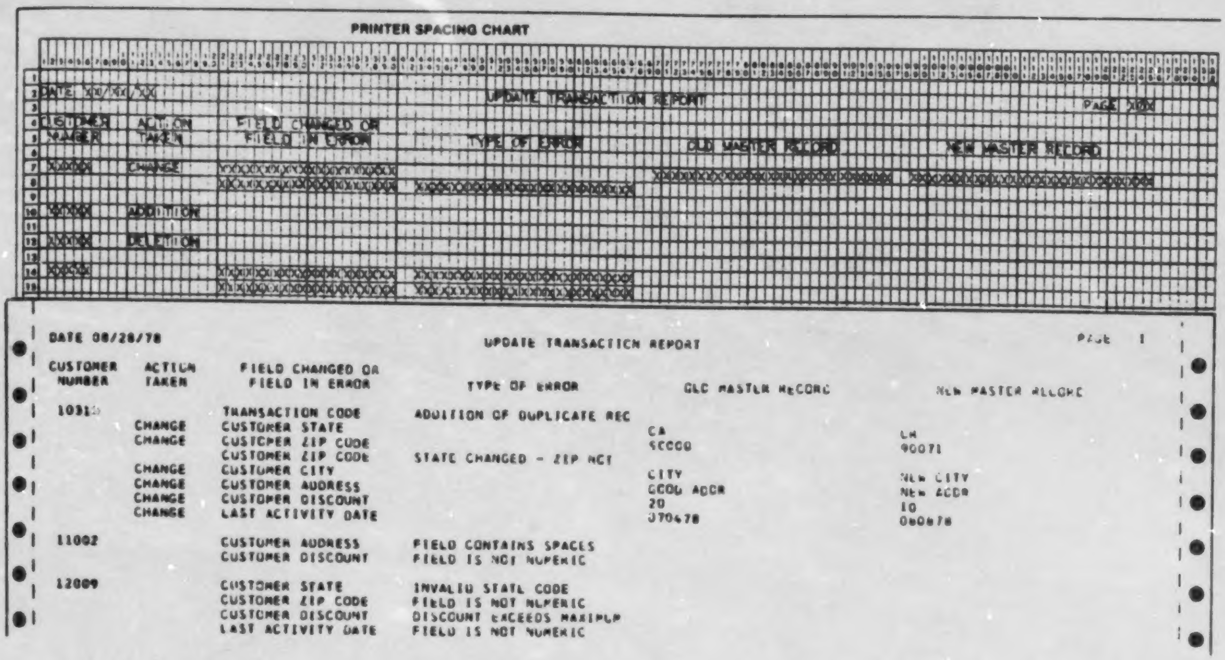
FIELD	POSITION	NUMBER OF DIGITS	ATTRIBUTES
Deletion Code	1	1	Alphanumeric
Customer Number (Key)	2 - 6	5	Alphanumeric
Customer Name	7 - 26	20	Alphanumeric
Customer Address	27 - 53	27	Alphanumeric
Customer City	54 - 73	20	Alphanumeric
Customer State	74 - 75	2	Alphanumeric
Customer Zip Code	76 - 80	5	Numeric
Customer Discount	81 - 82	2	Numeric
Last Activity Date	83 - 88	6	Numeric

File Information:
Record Length -- 88 characters
Blocking Factor -- 20 records per block
Storage Medium -- Disk
Organization -- Indexed Sequential
Access -- Random

Format of Customer Transaction Input File


CUST NUMBER
DISCOUNT
LAST ACTIVITY DATE
CODE
CUST. NUMBER
CUSTOMER NAME
CUSTOMER ADDRESS
CUSTOMER CITY
STATE
ZIP CODE
CODE

Format of Transaction Report



PRINTER SPACING CHART

```
 1
 2 DATE XX/XX/XX                                    UPDATE TRANSACTION REPORT                                                     PAGE XXX
 3
 4 CUSTOMER   ACTION     FIELD CHANGED OR
 5  NUMBER    TAKEN       FIELD IN ERROR            TYPE OF ERROR           OLD MASTER RECORD            NEW MASTER RECORD
 6
 7 XXXXX      CHANGE     XXXXXXXXXXXXXXXXXXXX                                XXXXXXXXXXXXXXXXXXXXXXXXXXXXX  XXXXXXXXXXXXXXXXXXXXXXXXXXXXX
 8                       XXXXXXXXXXXXXXXXXXXX  XXXXXXXXXXXXXXXXXXXXXXXXX
 9
10 XXXXX      ADDITION
11
12 XXXXX      DELETION
13
14 XXXXX                 XXXXXXXXXXXXXXXXXXXX  XXXXXXXXXXXXXXXXXXXXXXXXX
15                       XXXXXXXXXXXXXXXXXXXX  XXXXXXXXXXXXXXXXXXXXXXXXX
```

```
DATE 08/28/78                                 UPDATE TRANSACTION REPORT                                      PAGE   1

CUSTOMER   ACTION      FIELD CHANGED OR
 NUMBER    TAKEN        FIELD IN ERROR         TYPE OF ERROR             OLD MASTER RECORD         NEW MASTER RECORD

 1031[illegible]          TRANSACTION CODE      ADDITION OF DUPLICATE REC
           CHANGE      CUSTOMER STATE                                  CA                        LA
           CHANGE      CUSTOMER ZIP CODE                               90000                     90071
                       CUSTOMER ZIP CODE      STATE CHANGED - ZIP NOT
           CHANGE      CUSTOMER CITY                                   CITY                      NEW CITY
           CHANGE      CUSTOMER ADDRESS                                GOOD ADDR                 NEW ADDR
           CHANGE      CUSTOMER DISCOUNT                               20                        10
           CHANGE      LAST ACTIVITY DATE                              070478                    080678
 11002                 CUSTOMER ADDRESS       FIELD CONTAINS SPACES
                       CUSTOMER DISCOUNT      FIELD IS NOT NUMERIC

 12009                 CUSTOMER STATE         INVALID STATE CODE
                       CUSTOMER ZIP CODE      FIELD IS NOT NUMERIC
                       CUSTOMER DISCOUNT      DISCOUNT EXCEEDS MAXIMUM
                       LAST ACTIVITY DATE     FIELD IS NOT NUMERIC
```

BEST COPY AVAILABLE

535

INDEXED SEQUENTIAL FILES II

UNIT XIII

ASSIGNMENT SHEET 1: Understanding the Sample Program Specifications

1. Explain the purpose of this unit's sample program.

2. What are two of the reasons for random update of the Customer Master File?

3. Explain the purpose of each file used in the program.

INDEXED SEQUENTIAL FILES II

UNIT XIII

ANSWERS TO ASSIGNMENT SHEET

Assignment I

1. This program randomly updates the CUSTOMER MASTER FILE, which was created in the previous unit.

2. Some of the reasons are (any two):

   1. fast response
   2. better economic control of sales
   3. better customer service
   4. support of on-line shipping
   5. and other

3. Transaction file - inputs records to be used for updating the master file

   Customer Master File - the randomly updated customer file

   Permanent file - outputs customer printed lines for the update transaction report

INDEXED SEQUENTIAL FILE UPDATING II

UNIT XIII

JOB SHEET I: Writing a Complete Indexed Sequential Program for Payroll (with a Sample Solved Indexed Sequential Program for guidance).

Materials

This job is to be done in COBOL on you school's computer. As part of the assignment you will complete IPO charts, COBOL coding and test, debug and document an indexed sequential program for payroll. The payroll application is the same as in Unit XI in order to avoid the need to learn a new application.

Introduction

A simple but complete indexed sequential file update maintenance program for payroll is the application for this job. You are provided with a similar application of an indexed sequential file maintenance program for an inventory. The sample application; will greatly help you when designing and completing your own application, so study it carefully. At this time, turn to the appendix for this job sheet and carefully review the first program called VSAM SETUP Figure JS I-7. Then return to this page.

Overview of the Payroll Indexed Sequential Application

Figure JS I-I is an overview of the job. In Figure JS I-I notice that three programs (or program segments) are depicted. These are:

1. HI - utility program to create indexed sequential payroll master file (supplied by the computer manufacturer or instructor)
2. H2 - COBOL program to randomly update the indexed sequential payroll master file.
3. H3 - COBOL program to sequentially print the updated master in report format.

The first program is an indexed sequential file utility program, which uses the facilities supplied by IBM to create the file, its index, and to load the file with the contents of the test data. This program also deletes previous versions (after the first run of your program the file resulting from the run must be deleted so that test results are not confused with test results of previous runs) of the master file. This program is not written in COBOL but is in Job Control Language of IBM.)

Many different ways exist to accomplish similar functions of creating indexed sequential master files. Your school may use this method or others. Please check with you instructor to determine in which manner the master file will be created. You will use your instructor's method to formulate your own version of HI.)

H2 is the center of the random update activity. It reads transactions (from cards or disk as specified by your instructor) and retrieves the matching master record for the transaction, updates the master's contents and then (usually) rewrites it back in place. In the case of new master records it checks to see if there is not an existing master already in place - an error condition. Then it writes the record to add it to the file. In the case of deletes it simply issues a DELETE command (note: this is a different convention from that described in the information sheets for this unit but this is standard in the 1974 COBOL compilers.)

H3 simply reads the newly updated master file in sequential order (through special commands) and prints a brief report of its contents. (Note H3 may actually be a subroutine in H2, depending upon your design.)

Starting the Job - H1

You will start this job by obtaining a copy of the appropriate version of H1 from your instructor for your school. This job is ordinarily run each time before you run your H2 job but usually no COBOL programming may be required depending upon your school's version.

Completing H1

Check with your instructor for information and specifications to run H1 at your school. Ordinarily you will run H1 each time you ran H2.

Starting the JOB - H2

As in the previous job, you will only write the procedure division for program H2. Your H1 program (or utility) must be working successfully before starting to test H2, as testing H2 requires the indexed sequential master file from H1. At this time, please review the listing of the sample solved program in the appendix to this job sheet (JS 1-8).

Figure JS 1-2, Hierarchy Chart for Payroll Indexed Sequential File Update Program, H2, has been provided for your use in writing the program. Carefully review it before beginning to draw IPO charts for each module.

Figure JS 1-3, Pseudocode for Payroll Indexed Sequential File Update Program, H2 has also been provided to help you complete the program. You should complete the form by filling out the far right column in the form by reference to the three pre written divisions (Figure JS 1-4).

Completing H2

To complete H2 you should:

1. Draw IPO charts for each module.

2. Write pseudocode for any additional modules not included in the prewritten psuedocode; e.g., NEW-PAGE, REPORT-RTN, etc.

3. Code the program.

4. Test the program using the supplied test data from the instructor.

5. Check the program's output for correctness against Figure JS 1-5.

6. Hand in:

   a. IPO charts

   b. Pseudocode

   c. Coding sheets

   d. Successful COBOL run

Starting the Job - H3

The purpose of H3 is to print the indexed sequential master file's contents in sequential order and in report format. You should modify your H2 program to:

1. Use the ACCESS IS DYNAMIC clause in the SELECT sentence for the master file. (Your instructor may suggest alternative methods.)
2. Modify the closing routine to read and print the master file. Note: use the AT END clause with the sequential read in this module.
3. Close the files and stop.

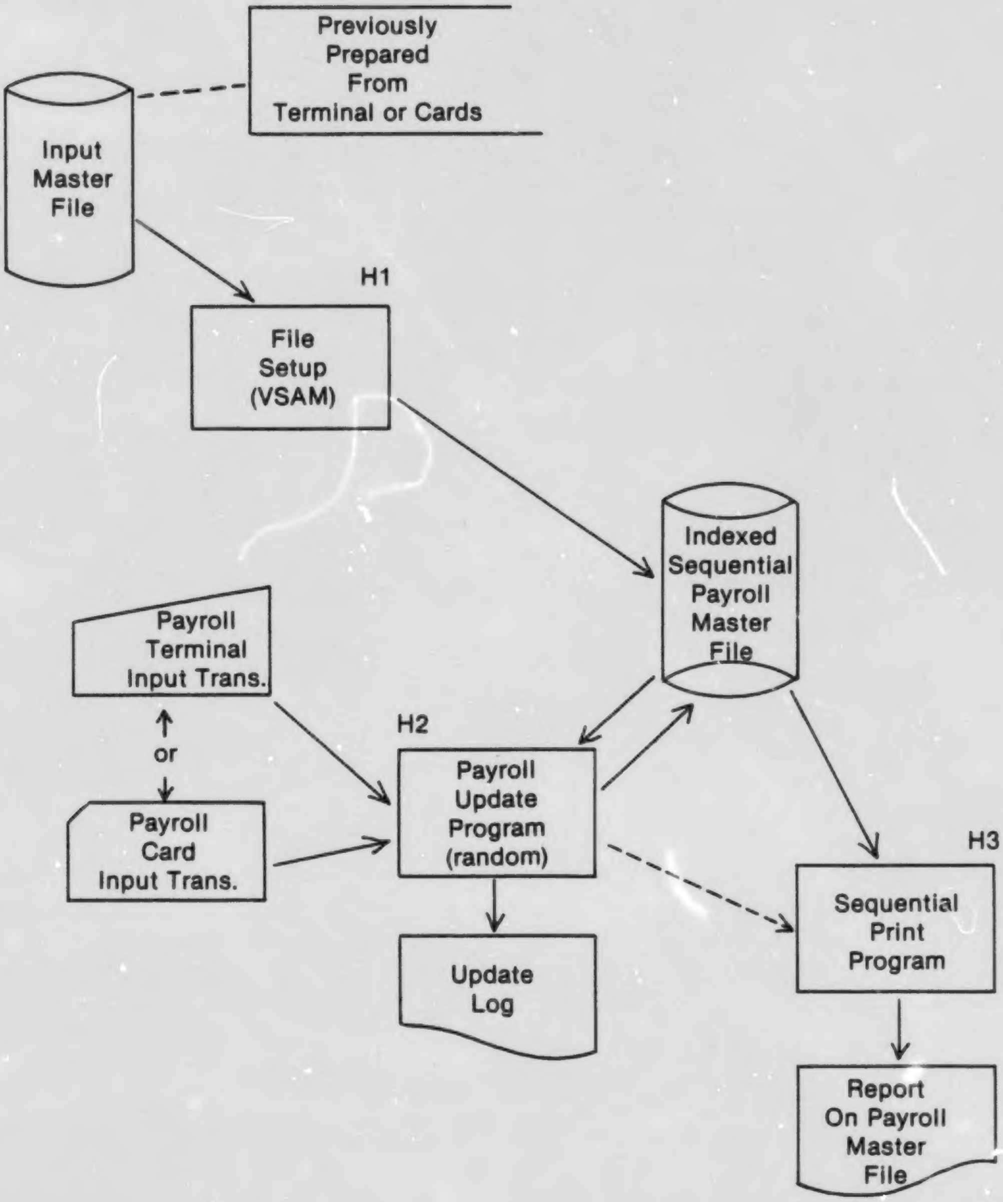
Design of the hierarchy chart, IPO, pseudocode, and coding is left to the student as this is a relatively small job.

Completing H3

To complete H3:

1. Draw a hierarchy chart.
2. Draw IPO charts.
3. Write pseudocode for each module.
4. Design report format and write COBOL code.
5. Test the program.
6. Check the program output against Figure JS I-6. (Note that the output was simply displayed and was not put into proper report format.)
7. Hand in:
   a. Hierarchy chart
   b. IPO chart
   c. Pseudocode
   d. Coding sheets
   e. Successful COBOL

Figure JS1-1 Overview of Application to Create, Update, and Print Indexed Sequential Payroll Master File


Previously Prepared From Terminal or Cards
Input Master File
H1
File Setup (VSAM)
Indexed Sequential Payroll Master File
Payroll Terminal Input Trans.
or
Payroll Card Input Trans.
H2
Payroll Update Program (random)
Update Log
H3
Sequential Print Program
Report On Payroll Master File

Note: H2 and H3 may be parts of the same program.

Figure JS1-2 Hierarchy Chart for Payroll Indexed Sequential File Update Program, H2

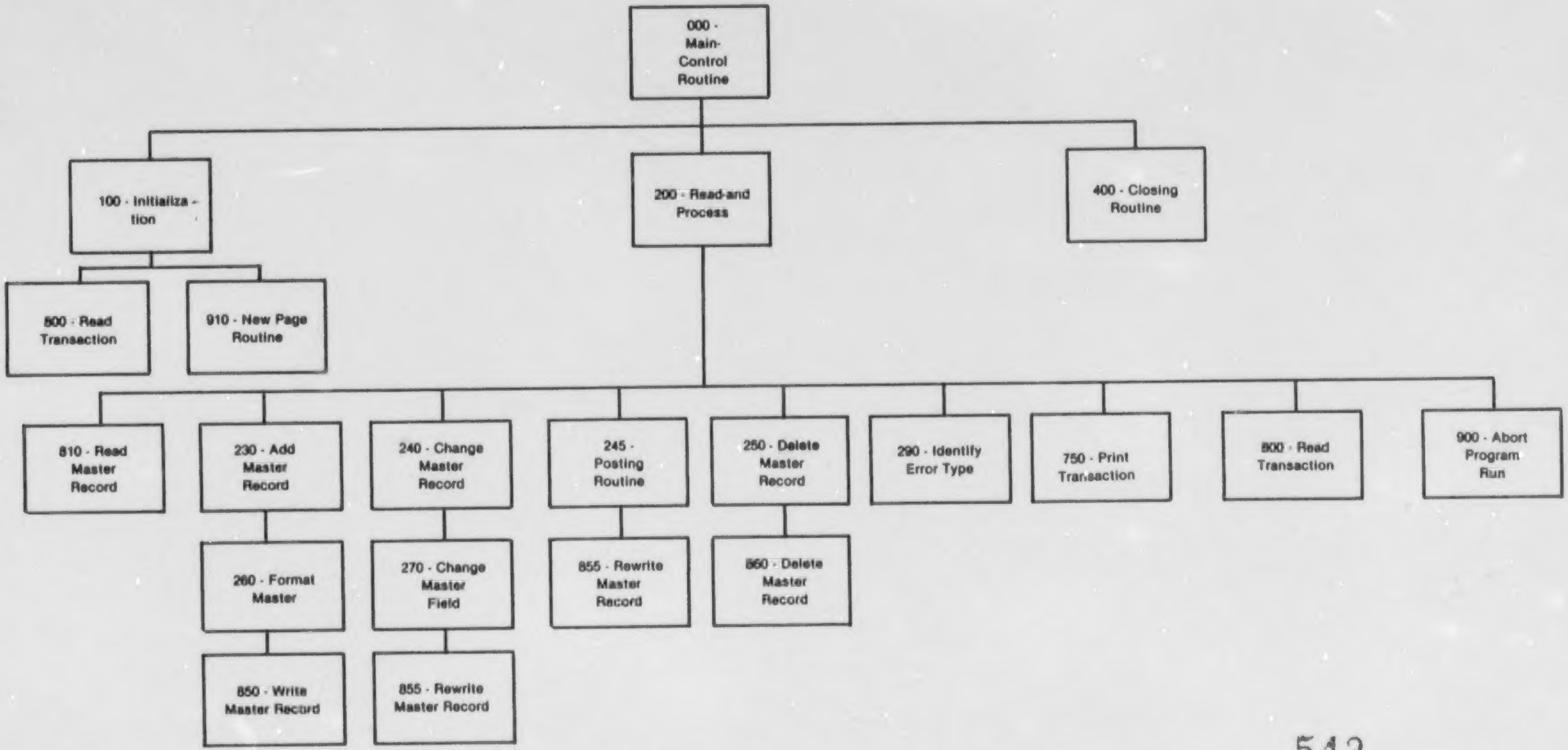

000 - Main-Control Routine
100 - Initializa-tion
200 - Read-and Process
400 - Closing Routine
800 - Read Transaction
910 - New Page Routine
810 - Read Master Record
230 - Add Master Record
240 - Change Master Record
245 - Posting Routine
250 - Delete Master Record
290 - Identify Error Type
750 - Print Transaction
800 - Read Transaction
900 - Abort Program Run
260 - Format Master
270 - Change Master Field
855 - Rewrite Master Record
860 - Delete Master Record
850 - Write Master Record
855 - Rewrite Master Record

Figure JS1-3 Pseudocode Indexed Sequential File Update Program, H2 (Part 1 of 5)

Pseudocode Specifications

DATE: June 17	PROGRAMMER: Jane Austin	PROGRAM: Payroll
MODULE FUNCTION:	REF:	MODULE NAME:

PSEUDOCODE	REF:	FILES, RECORDS, FIELDS REQUIRED

```
MAIN-CONTROL-ROUTINE
   DO INITIALIZATION
   DO READ-AND-PROCESS UNTIL
         END-OF-JOB.
   DO CLOSING
   STOP

INITIALIZATION
   OPEN I-O MASTER FILE
   OPEN OUTPUT PRINT FILE
   DO READ TRANSACTION
   DO NEW PAGE

READ-AND-   PROCESS
   CLEAR DETAIL LINE
   IF VALID UPDATE CODE
      DO READ MASTER.
   IF ADDITION AND NO MASTER
                  RECORD FOUND
      SET PROGRAM ABEND SWITCH OFF
      DO ADD MASTER
   ELSE
      IF CHANGE AND SUCCESSFUL
                  COMPLETION
         DO CHANGE MASTER RECORD
      ELSE
         IF POSTING AND SUCCESSFUL
                     COMPLETION
            DO POSTING RTN
         ELSE
            IF DELETION AND SUCCESSFUL
                     COMPLETION
               DO DELETION RTN
            ELSE
               DO ERROR RTN.
   DO PRINT TRANSACTION
   IF NOT PROGRAM ABEND
```

Figure JS1-3 Pseudocode Indexed Sequential File Update Program, H2 (Part 2 of 5)

Pseudocode Specifications

DATE: June 17	**PROGRAMMER:** Jane Austin	**PROGRAM:** Payroll
MODULE FUNCTION:	**REF:**	**MODULE NAME:**

PSEUDOCODE	REF:	FILES, RECORDS, FIELDS REQUIRED

```
        DO READ TRANSACTION
    ELSE
        DO PROGRAM ABORT.

ADD MASTER RECORD
    DO FORMAT MASTER RECORD
    DO WRITE MASTER RECORD
    IF NOT PROGRAM ABEND
        SET ACTION TO 'ADDED'.

CHANGE MASTER RECORD
    DO CHANGE MASTER FIELDS
    DO REWRITE MASTER RECORD
    IF NOT PROGRAM ABEND
            SET ACTION TO 'CHANGED'.

POSTING ROUTINE
    IF GROSS PAY NUMERIC
        ADD GROSS PAY TO TOTAL
        ADD GROSS PAY TO OLD GROSS.
    IF NET NUMERIC
        ADD NET TO TOTAL AND OLD.
    DO REWRITE MASTER

DELETE MASTER
    DO DELETE MASTER RECORD
    IF NOT PROGRAM ABEND
        SET ACTION TO 'DELETED'.

FORMAT MASTER RECORD
    MOVE FIELDS TO MASTER

CHANGE MASTER FIELDS
    MOVE NON-BLANK FIELDS TO
            MASTER
```

Figure JS1-3 Pseudocode Indexed Sequential File Update Program, H2 (Part 3 of 5)

Pseudocode Specifications

DATE: June 17
PROGRAMMER: Jane Austin
PROGRAM: Payroll
MODULE FUNCTION:
REF:
MODULE NAME:

```
PSEUDOCODE                              REF:    FILES, RECORDS, FIELDS
                                                REQUIRED

ERROR ROUTINE
   IDENTIFY ERROR TYPE AND
   SET ACTION TO ERROR REASON

CLOSING ROUTINE
   CLOSE FILES

PRINT TRANSACTION
   FORMAT TRANSACTION DETAIL LINE
   CHECK LINE COUNT FOR OVERFLOW
   PRINT DETAIL LINE

READ TRANSACTION
   READ TRANSACTION FILE
   IF TRAILER RECORD
      SET END OF JOB SWITCH ON.
   DISPLAY TRANSACTION                   D

READ MASTER
   SET UP KEY FOR INDEX SEARCH
   SET 'READ' IN ACTION FIELD
   READ MASTER
      INVALID KEY SET ABEND SWITCH
            ON.
   IF NOT SUCCESSFUL COMPLETION
      SET ABEND SWITCH ON.
   DISPLAY STATUS, OPERATION AND         D
            MASTER CONTENTS
```

Figure JS1-3 Pseudocode Indexed Sequential File Update Program, H2 (Part 4 of 5)

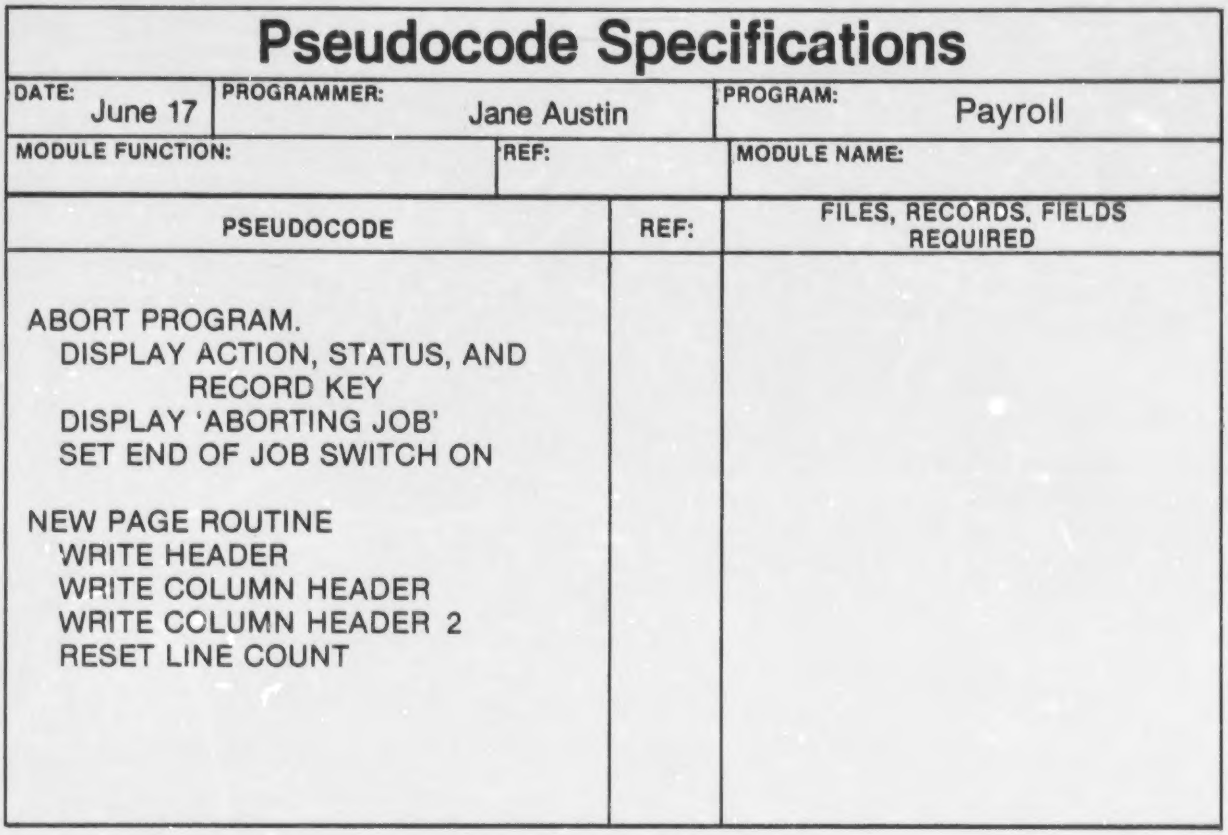
Pseudocode Specifications

DATE: June 17	PROGRAMMER: Jane Austin	PROGRAM: Payroll
MODULE FUNCTION:	REF:	MODULE NAME:

PSEUDOCODE	REF:	FILES, RECORDS, FIELDS REQUIRED

```
WRITE MASTER
SET UP KEY FOR INDEX SEARCH
SET 'WRITE' IN ACTION FIELD
WRITE MASTER
    INVALID KEY SET ABEND SWITCH
        ON.
IF NOT SUCCESSFUL COMPLETION
    SET ABEND SWITCH ON.
DISPLAY STATUS, OPERATION AND              D
        MASTER CONTENTS

DELETE MASTER
    SET UP KEY FOR INDEX SEARCH
    SET 'DELETE' IN ACTION FIELD
    DELETE MASTER
        INVALID KEY SET ABEND SWITCH
            ON.
    IF NOT SUCCESSFUL COMPLETION
        SET ABEND SWITCH ON.
    DISPLAY STATUS, OPERATION AND          D
            MASTER CONTENTS

REWRITE MASTER
    SET ACTION FIELD TO 'REWRITE'
    SET KEY FOR INDEXED ACCESS
    REWRITE MASTER
        INVALID KEY SET ABEND SWITCH
              ON.
    IF NOT SUCCESSFUL COMPLETION
        SET ABEND SWITCH ON.
    DISPLAY ACTION. STATUS AND
              MASTER CONTENTS
```

Figure JS1-3 Pseudocode Indexed Sequential File Update Program, H2 (Part 5 of 5)



Pseudocode Specifications

DATE: June 17	PROGRAMMER: Jane Austin	PROGRAM: Payroll
MODULE FUNCTION:	REF:	MODULE NAME:

PSEUDOCODE	REF:	FILES, RECORDS, FIELDS REQUIRED

```
ABORT PROGRAM.
   DISPLAY ACTION, STATUS, AND
               RECORD KEY
   DISPLAY 'ABORTING JOB'
   SET END OF JOB SWITCH ON

NEW PAGE ROUTINE
   WRITE HEADER
   WRITE COLUMN HEADER
   WRITE COLUMN HEADER 2
   RESET LINE COUNT
```

Note: lines with "D" in reference column should be displayed during the debugging of the program to aid in finding errors.

Figure JS1-4 Prewritten Divisions for Indexed Sequential Payroll Application (Part 1 of 2)

```
*
 IDENTIFICATION DIVISION.
*
 PROGRAM-ID. PAYROL.

 ENVIRONMENT DIVISION.
*
 CONFIGURATION SECTION.
 SOURCE-COMPUTER. IBM-370.
 OBJECT-COMPUTER. IBM-370.
*
 SPECIAL-NAMES.
     C01 IS TO-TOP-OF-PAGE.
*
 INPUT-OUTPUT SECTION.
 FILE-CONTROL.
     SELECT PRINT-FILE ASSIGN TO UR-S-SYSPRINT.
     SELECT  TRANSACTION-FILE ASSIGN TO UR-S-SYSIN.
     SELECT OLD-MASTER-FILE ASSIGN TO DA-UMCSC1
            ORGANIZATION IS INDEXED
            ACCESS MODE IS RANDOM
            RECORD KEY IS KEY-EMPLOYEE-ID
            FILE STATUS IS FILE-STATUS-FLAG.
 DATA DIVISION.
*
  FILE SECTION.
 FD  OLD-MASTER-FILE
     LABEL RECORD IS OMITTED.
 01  OLD-MASTER-RECORD.
     05  FILLER                        PIC X(25).
     05  KEY-EMPLOYEE-ID               PIC X(10).
     05  FILLER                        PIC X(45).
*
 FD   TRANSACTION-FILE
     LABEL RECORD IS OMITTED.
 01  TRANSACTION-RECORD         PIC X(80).
*
 FD  PRINT-FILE
     LABEL RECORD IS OMITTED.
 01  PRINT-LINE                     PICTURE X(132).
*
 WORKING-STORAGE SECTION.
 01  LINE-LIMIT                     PIC  9(02) VALUE 30.
 01  LINE-COUNT                     PIC 9(01) VALUE 0.
 01  ERROR-FLAG                     PIC X(01) VALUE 'N'.
 01  END-OF-JOB-SWITCH              PIC X(01) VALUE 'N'.
     88  END-OF-JOB                             VALUE 'Y'.
 01  WS-OLD-MASTER-RECORD.
     05  OLD-EMPLOYEE-NAME           PIC X(25).
     05  OLD-EMPLOYEE-ID             PIC X(10).
     05  OLD-DEPARTMENT              PIC X(15).
     05  OLD-PAY-RATE                PIC 99V99.
     05  OLD-GROSS-PAY-YTD           PIC 99999V99.
     05  OLD-NET-PAY-YTD             PIC 99999V99.
PAYROL       11.41.35            NOV 19,1983
     05  FILLER                      PIC X(12).
*
 01  WS-TRANSACTION-RECORD.
     05  EMPLOYEE-NAME                PIC X(25).
     05  EMPLOYEE-ID                  PIC X(10).
     05  DEPARTMENT                    PIC X(15).
     05  PAY-RATE                PIC 99V99.
     05  GROSS-PAY                     PIC 9999V99.
     05  NET-PAY                     PIC 9999V99.
     05  FILLER                      PIC X(11).
```

```
     05  TRANSACTION-TYPE                   PIC XXX.
         88  ADDITION                                  VALUE 'ADD'.
         88  DELETION                                  VALUE 'DEL'.
         88  CHANGE                                    VALUE 'CHG'.
         88  POSTING                                   VALUE 'PST'.
         88  VALID-UPDATE-CODE                         VALUE 'ADD'
                                                             'DEL'
 01  SWITCHES.                                               'CHG'
     05  PROGRAM-ABEND-SWITCH       PIC X VALUE 'N'.         'PST'.
         88  PROGRAM-ABEND                VALUE 'Y'.
     05  FILE-STATUS-FLAG           PIC XX VALUE '00'.
         88  SUCCESSFUL-COMPLETION       VALUE '00'.
         88  DUPLICATE-ALTERNATE-KEY               VALUE '02'.
         88  END-OF-FILE                           VALUE  '10'.
         88  SEQUENCE-ERROR                        VALUE '21'.
         88  DUPLICATE-KEY                         VALUE '22'.
         88  NO-RECORD-FOUND                       VALUE '23'.
         88  BOUNDARY-VIOLATION-IND-REL            VALUE '24'.
         88  PERMANENT-I-O-DATA-ERROR              VALUE '30'.
         88  BOUNDARY-VIOLATION-SEQ                VALUE '34'.
         88  LOGIC-ERROR                           VALUE '92'.
     05  I-O-OPERATION-FLAG       PIC X(07) VALUE '       '.
*
 01  REPORT-TITLE.
     05  FILLER                     PICTURE X(50)   VALUE SPACES.
     05  FILLER                     PICTURE X(33)   VALUE
            'PAYROLL UPDATE TRANSACTION REPORT'.
     05  FILLER                     PICTURE X(49)   VALUE SPACES.
*
 01  HEADING-LINE-1.
     05  FILLER                     PICTURE X(4)    VALUE SPACES.
     05  FILLER                     PICTURE X(5)    VALUE 'TRANS'
     05  FILLER                     PICTURE X(72)   VALUE SPACES.
     05  FILLER                     PICTURE X(9)    VALUE
                          'RATE     '.
     05  FILLER                     PICTURE X(20)   VALUE
                          'GROSS PAY     NET PAY'.
     05  FILLER                     PICTURE X(22)   VALUE SPACES.
*
 01  HEADING-LINE-2.
     05  FILLER                     PICTURE X(4)    VALUE SPACES.
     05  FILLER                     PICTURE X(4)    VALUE 'CODE'.
     05  FILLER                     PICTURE X(5)    VALUE SPACES.
     05  FILLER                     PICTURE X(15)   VALUE
                          'ID NUMBER      '.
PAYROL       11.41.35            NOV 19,1983
     05  FILLER                     PICTURE X(30)   VALUE
                          'EMPLOYEE NAME                 '.
     05  FILLER                     PICTURE X(22)   VALUE
                          'DEPARTMENT           '.
     05  FILLER                     PICTURE X(30)   VALUE
                          'OF PAY      THIS PD       THIS PD'.
     05  FILLER                     PICTURE X(22)   VALUE
                          '  CAUSE OF ERRORS     '.
*
 01  DETAIL-LINE.
     05  FILLER                     PICTURE X(5)    VALUE SPACES.
     05  DL-TRANSACTION-TYPE                  PIC XXX.
     05  FILLER                     PICTURE X(5)    VALUE SPACES.
     05  DL-EMPLOYEE-ID                  PIC X(10).
     05  FILLER                     PICTURE X(5)    VALUE SPACES.
     05  DL-EMPLOYEE-NAME                PIC X(25).
     05  FILLER                     PICTURE X(5)    VALUE SPACES.
     05  DL-DEPARTMENT                     PIC X(15).
     05  FILLER                     PICTURE X(7)    VALUE SPACES.
     05  DL-PAY-RATE                PIC $$9.99.
     05  FILLER                     PICTURE X(5)    VALUE SPACES.
     05  DL-GROSS-PAY                    PIC $$$$.99.
     05  FILLER                     PICTURE X(5)    VALUE SPACES.
     05  DL-NET-PAY                   PIC $$$$.99.
     05  FILLER                     PICTURE X(2)    VALUE SPACES.
```

Figure JS1-4 Prewritten Divisions for Indexed Sequential Payroll Application (Part 2 of 2)

```
*
 01  IO-ERROR-LINE REDEFINES DETAIL-LINE.
     05  FILLER                  PIC X(01).
     05  FILLER                  PIC X(01).
     05  IO-FILE-STATUS-FLAG     PIC X(02).
     05  FILLER                  PIC X(01).
     05  IO-OPERATION-FLAG       PIC X(07).
     05  FILLER                  PIC X(01).
     05  IO-ERROR-KEY            PIC X(10).
     05  FILLER                  PIC X(109).
 01  TOTAL-LINE.
     05  FILLER                     PICTURE X(10)   VALUE SPACES.
     05  FILLER                     PICTURE X(19)   VALUE
                         TOTAL TRANSACTIONS:'.
     05  TOTAL-LINE-TRANS                PIC ZZZZ9.
     05  FILLER                     PICTURE X(15)   VALUE
                              '     CORRECT:'.
     05  TOTAL-LINE-RIGHT                PIC ZZZZ9.
     05  FILLER                     PICTURE X(14)   VALUE
                              '     ERRORS:'.
     05  TOTAL-LINE-WRONG                PIC ZZZZ9.
```

```
     05  FILLER                     PICTURE X(23)   VALUE
                                    TOTAL GROSS PAY:'.
     05  TOTAL-LINE-GROSS-PAY               PIC $$$$$9.99.
     05  FILLER                     PICTURE X(10)   VALUE
                              '  NET PAY:'.
     05  TOTAL-LINE-NET-PAY                 PIC $$$$$9.99.
     05  FILLER                     PICTURE X(6)    VALUE SPACES.
*
 01  TOTALS.
     05  TOTAL-TRANS            PIC 9(04) VALUE ZERO.
     05  TOTAL-RIGHT            PIC 9(04) VALUE ZERO.
     05  TOTAL-WRONG            PIC 9(04) VALUE ZERO.
     05  TOTAL-GROSS-PAY        PIC 9(07)V99  VALUE ZERO.
     05  TOTAL-NET-PAY          PIC 9(07)V99  VALUE ZERO.
 01  TEMP-MAST-STORE            PIC X(80).
*
```

BEST COPY AVAILABLE

Figure JS1-5 Output from Program H2

```
                                          PAYROLL UPDATE TRANSACTION REPORT

TRANS                                                                       RATE     GROSS PAY      NET PAY
CODE      ID NUMBER    EMPLOYEE NAME                  DEPARTMENT           OF PAY     THIS PD      THIS PD  UPDATE ACTION

 PAY      BUR-145927                                                        $0.00     $158.00      $126.40  POSTED

 CHG      BUR-382196   BURDEN, JOHN                                         $0.00        $.00         $.00  CHANGED

 ADD      CAL-336741   CALVERT, CATHLEEN              MARKETING             $6.25        $.00         $.00  ADDED

 DEL      FAV-835127                                                        $0.00     $190.00      $152.00  DELETED

 PAY      FIN-571433                                                        $0.00     $190.00      $152.00  POSTED

 CHG      HAM-446735                                  ACCOUNTING            $0.00        $.00         $.00  CHANGED

 CHG      MAS-619318                                  PRODUCTION            $5.00        $.00         $.00  CHANGED

 PAY      MAS-619318                                                        $0.00     $200.00      $160.00  POSTED

 DAL      POR-783914                                                        $0.00        $.00         $.00

 CHG      ROB-445331                                                        $4.99     $318.00      $254.00  CHANGED

 PAY      SLA-108423                                                        $0.00     $374.00      $280.50  POSTED

 PAY      WAT-623489                                                        $0.00     $250.00      $205.73  POSTED

 ADD      WIL-300177   WILKES, ASHLEY                 ACCOUNTING            $8.35        $.00         $.00  ADDED



TOTAL TRANSACTIONS:    13      CORRECT:    13      ERRORS:     1      TOTAL GROSS PAY: $1172.00  NET PAY:  $924.63
```

Figure JS1-6 Output Showing Updating and Debugging Traces (Part 1 of 2)

```
   READ-M = BURKE, SADIE             BUR-145927DATA PROCESSING039501580000126400
   STATUS = 00
OPERATION = READ
   STATUS = 00
OPERATION = REWRITE
            BURKE, SADIE             BUR-145927DATA PROCESSING039501738000139040
   READ-M = BURDEN, JACK             BUR-382196ACCOUNTING     073502940000220500
   STATUS = 00
OPERATION = READ
   STATUS = 00
OPERATION = REWRITE
            BURDEN, JOHN             BUR-382196ACCOUNTING     073502940000220500
   READ-M = BURDEN, JOHN             BUR-382196ACCOUNTING     073502940000220500
   STATUS = 23
OPERATION = READ
   STATUS = 00
OPERATION = WRITE
            CALVERT, CATHLEEN        CAL-336741MARKETING      06250     00     0
   READ-M = FAVIR, LARRY             FAV-835127MARKETING      070002800000210000
   STATUS = 00
OPERATION = READ
   STATUS = 00
OPERATION = DELETE
            FAVIR, LARRY             FAV-835127MARKETING      070002800000210000
   READ-M = FINKBINE, GEORGE         FIN-571433ACCOUNTING     047501900000150000
   STATUS = 00
OPERATION = READ
   STATUS = 00
OPERATION = REWRITE
            FINKBINE, GEORGE         FIN-571433ACCOUNTING     047502090000165200
   READ-M = HAMILTON, MELANIE        HAM-446735DATA PROCESSING038501732500138600
   STATUS = 00
OPERATION = READ
   STATUS = 00
OPERATION = REWRITE
            HAMILTON, MELANIE        HAM-446735ACCOUNTING     038501732500138600
   READ-M = MASTERN, CASS            MAS-619318MARKETING      045001912500153000
   STATUS = 00
OPERATION = READ
   STATUS = 00
OPERATION = REWRITE
            MASTERN, CASS            MAS-619318PRODUCTION     050001912500153000
   READ-M = MASTERN, CASS            MAS-619318PRODUCTION     050001912500153000
   STATUS = 00
OPERATION = READ
   STATUS = 00
OPERATION = REWRITE
            MASTERN, CASS            MAS-619318PRODUCTION     050002112500169000
UNABLE UPDATE - INVALID CODE
   READ-M = ROBILLARD, ELLEN         ROB-445331DATA PROCESSING036001440000115200
   STATUS = 00
OPERATION = READ
   STATUS = 00
OPERATION = REWRITE
            ROBILLARD, ELLEN         ROB-445331DATA PROCESSING049900318000025400
   READ-M = SLATTERY, THOMAS         SLA-108423ACCOUNTING     093503740000280500
   STATUS = 00
OPERATION = READ
   STATUS = 00
```

Figure JS1-6 Output Showing Updating and Debugging Traces (Part 2 of 2)

```
OPERATION = REWRITE
            SLATTERY, THOMAS         SLA-108423ACCOUNTING     093504114000308550
   READ-M = WATLING, BELLE           WAT-623489MARKETING      063202528000189600
   STATUS = 00
OPERATION = READ
   STATUS = 00
OPERATION = REWRITE
            WATLING, BELLE           WAT-623489MARKETING      063202778000210173
   READ-M = WATLING, BELLE           WAT-623489MARKETING      063202778000210173
   STATUS = 23
OPERATION = READ
   STATUS = 00
OPERATION = WRITE
            WILKES, ASHLEY           WIL-300177ACCOUNTING     08350     00      0
   STATUS = 22
OPERATION = WRITE
            WILKES, ASHLEY           WIL-300177ACCOUNTING     08350     00      0
END OF PAYROLL MASTER FILE UPDATE PROGRAM
```

APPENDIX

Job Sheet 1: Sample Application Indexed Sequential Update Program for Inventory

(See Figure JS 1-7 and Figure JS 1-8 on the following pages.)

Figure JS1-7 VSAM Setup Program for an Inventory Application (Part 1 of 2)

```
 1     //VSAM1 JOB (XXXXLH),'VSAM TEST',                                    JOB 1503
       //      PASSWORD=,
       //      MSGLEVEL=(1,1)
 2     //PURG EXEC PGM=IDCAMS
 3     //SYSPRINT  DD  SYSOUT=A
 4     //SYSIN     DD  *
 5     //STEP1  EXEC PGM=IDCAMS
 6     //SYSPRINT   DD  SYSOUT=A
 7     //SYSIN      DD  *
 8     //STEP2   EXEC PGM=IDCAMS
 9     //SYSPRINT  DD  SYSOUT=A
10     //IN        DD  DSN=$0819TK.PUBLIC.MASTER.FILE(MAST),DISP=SHR
11     //OUT       DD  DSN=ZSYS713.USER.X0819.ANCY,DISP=SHR
12     //SYSIN     DD  *


IEF236I ALLOC. FOR VSAM1 PURG
IEF237I JES2 ALLOCATED TO SYSPRINT
IEF237I JES2 ALLOCATED TO SYSIN
IEF237I 4FB  ALLOCATED TO SYS00001
IEF142I VSAM1 PURG - STEP WAS EXECUTED - COND CODE 0000
IEF285I   JES2.JOB01503.S00104                          SYSOUT
IEF285I   JES2.JOB01503.SI0101                          SYSIN
IEF285I   ZSYS713                                       KEPT
IEF285I   VOL SER NOS= SYS713.
IEF373I STEP /PURG    / START 83277.2231
IEF374I STEP /PURG    / STOP  83277.2231 CPU    0MIN 00.52SEC SRB    0MIN 00.03SEC VIRT   212K SYS   276K
IEF236I ALLOC. FOR VSAM1 STEP1
IEF237I JES2 ALLOCATED TO SYSPRINT
IEF237I JES2 ALLOCATED TO SYSIN
IEF237I 4FB  ALLOCATED TO SYS00002
IEF142I VSAM1 STEP1 - STEP WAS EXECUTED - COND CODE 0000
IEF285I   JES2.JOB01503.S00105                          SYSOUT
IEF285I   JES2.JOB01503.SI0102                          SYSIN
IEF285I   ZSYS713                                       KEPT
IEF285I   VOL SER NOS= SYS713.
IEF373I STEP /STEP1   / START 83277.2231
IEF374I STEP /STEP1   / STOP  83277.2232 CPU    0MIN 00.41SEC SRB    0MIN 00.03SEC VIRT   292K SYS   276K
IEF236I ALLOC. FOR VSAM1 STEP2
IEF237I JES2 ALLOCATED TO SYSPRINT
IEF237I 585  ALLOCATED TO IN
IEF237I 4FB  ALLOCATED TO SYS00506
IEF237I 4FB  ALLOCATED TO OUT
IEF237I 4FB  ALLOCATED TO SYS00508
IEF237I JES2 ALLOCATED TO SYSIN
IEF142I VSAM1 STEP2 - STEP WAS EXECUTED - COND CODE 0000
IEF285I   JES2.JOB01503.S00106                          SYSOUT
IEF285I   $0819TK.PUBLIC.MASTER.FILE                    KEPT
IEF285I   VOL SER NOS= SYS710.
IEF285I   SYSCTLG.VSYS713                               KEPT
IEF285I   VOL SER NOS= SYS713.
IEF285I   ZSYS713.USER.X0819.ANCY                       KEPT
IEF285I   VOL SER NOS= SYS713.
IEF285I   ZSYS713                                       KEPT
IEF285I   VOL SER NOS= SYS713.
IEF285I   JES2.JOB01503.SI0103                          SYSIN
IEF373I STEP /STEP2   / START 83277.2232
IEF374I STEP /STEP2   / STOP  83277.2232 CPU    0MIN 00.26SEC SRB    0MIN 00.02SEC VIRT   252K SYS   248K
IEF375I  JOB /VSAM1   / START 83277.2231
IEF376I  JOB /VSAM1   / STOP  83277.2232 CPU    0MIN 01.19SEC SRB    0MIN 00.08SEC
```

Figure JS1-7 VSAM Setup Program for an Inventory Application (Part 2 of 2)

```
IDCAMS  SYSTEM SERVICES                                   TIME: 22:31:51        10/04/83     PAGE    1

  DELETE (ZSYS713.USER.X0819.ANCY)

IDC0550I ENTRY (D) VSAMDSET.T94F9070.DFD83277.T963ECC2.T94F9070 DELETED

IDC0550I ENTRY (I) VSAMDSET.T94F9540.DFD83277.T963ECC2.T94F9540 DELETED

IDC0550I ENTRY (C) ZSYS713.USER.X0819.ANCY DELETED

IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0


IDC0002I IDCAMS PROCESSING COMPLETE. MAXIMUM CONDITION CODE WAS 0



IDCAMS  SYSTEM SERVICES                                   TIME: 22:31:56        10/04/83     PAGE    1

   DEFINE CLUSTER (NAME(ZSYS713.USER.X0819.ANCY) -
          KEYS (5,1) -
          VOLUME (SYS713) -
          RECORDSIZE (80,80) -
          FREESPACE (20,10) -
          RECORDS (50,10)) -
   CATALOG (ZSYS713/        )

IDC0508I DATA ALLOCATION STATUS FOR VOLUME SYS713 IS 0

IDC0509I INDEX ALLOCATION STATUS FOR VOLUME SYS713 IS 0

IDC0512I NAME GENERATED-(D) VSAMDSET.T82B1AF0.DFD83277.T963ECCE.T82B1AF0

IDC0512I NAME GENERATED-(I) VSAMDSET.T82B1FE0.DFD83277.T963ECCE.T82B1FE0

IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0


IDC0002I IDCAMS PROCESSING COMPLETE. MAXIMUM CONDITION CODE WAS 0



IDCAMS  SYSTEM SERVICES                                   TIME: 22:32:04        10/04/83     PAGE    1

  REPRO -
  INFILE(IN) -
  OUTFILE(OUT)

IDC0005I NUMBER OF RECORDS PROCESSED WAS 25

IDC0001I FUNCTION COMPLETED, HIGHEST CONDITION CODE WAS 0


IDC0002I IDCAMS PROCESSING COMPLETE. MAXIMUM CONDITION CODE WAS 0
```

Figure JS1-8 Listing of Sample Indexed Sequential File Update Program for Inventory (Part 1 of 7)

© The Curators of the
University of Missouri
1984, All Rights Reserved

```
          12.07.21        DEC  8,1983

 IDENTIFICATION DIVISION.
 PROGRAM-ID. ISINVEN.
 AUTHOR. CHUCK LITECKY
ENVIRONMENT DIVISION.
*
CONFIGURATION SECTION.
SOURCE-COMPUTER. IBM-370.
OBJECT-COMPUTER. IBM-370.
*
SPECIAL-NAMES.
    C01 IS TO-TOP-OF-PAGE.
*
INPUT-OUTPUT SECTION.
FILE-CONTROL.
    SELECT INVEN-TRANSACTION-FILE ASSIGN TO UR-S-SYSIN.
    SELECT INVEN-OLD-MASTER-FILE ASSIGN TO DA-UMCSCR1
           ORGANIZATION IS INDEXED
           ACCESS MODE IS RANDOM
           RECORD KEY IS IS-PART-NO
           FILE STATUS IS FILE-STATUS-FLAG.
    SELECT PRINT-FILE ASSIGN TO UR-S-SYSPRINT.
*
DATA DIVISION.
*
FILE SECTION.
FD INVEN-OLD-MASTER-FILE
    LABEL RECORD IS STANDARD.
01  INVEN-OLD-MASTER-RECORD.
    05  FILLER                          PIC X(01).
    05  IS-PART-NO                      PIC X(05).
    05  FILLER                          PIC X(74).
*
FD INVEN-TRANSACTION-FILE
    LABEL RECORD IS STANDARD.
01  INVEN-TRANSACTION-RECORD           PIC X(80).
*
FD PRINT-FILE
    LABEL RECORD IS STANDARD.
01  PRINT-LINE                         PIC X(132).
*
WORKING-STORAGE SECTION.
01  LINE-COUNT                 PIC 99       VALUE 99.
01  END-OF-JOB-SWITCH          PIC X        VALUE 'N'.
    88  END-OF-JOB                          VALUE 'Y'.
01  ERROR-FLAG                 PIC X        VALUE 'N'.
```

```
01  INV-VAL                    PIC 9(9)V99 VALUE 0.
01  T-INV-VAL                  PIC 9(10)V99 VALUE 0.
01  LINE-CNT                   PIC 99 VALUE 0.
01  PAGE-CNT                   PIC 99 VALUE 0.
*
01  WS-OLD-MAST.
    02  M-IDENT-CODE        PIC X.
    02  M-PART-NUM      PIC X(5).
    02  M-DESCRIPTION   PIC X(30).
    02  M-AMT-ON-HAND   PIC 9(5).
    02  M-REORDER-PT    PIC 9(5).
    02  M-OPTIMUM-LVL   PIC 9(5).
    02  M-MAX-QUAN      PIC 9(5).
    02  M-PRICE-PER     PIC 9(4)V99.
    02  FILLER          PIC X(18).
01  WS-TRAN.
    02  IDENT-CODE        PIC X.
    02  PART-NUM      PIC X(5).
    02  DESCRIPTION   PIC X(30).
    02  AMT-ON-HAND   PIC 9(5).
    02  REORDER-PT    PIC 9(5).
    02  OPTIMUM-LVL   PIC 9(5).
    02  MAX-QUAN      PIC 9(5).
    02  PRICE-PER     PIC 9(4)V99.
    02  TRANS-CODE    PIC X(3).
    88  DELETE-TRANSACTION  VALUE 'DEL'.
    88  ADD-TRANSACTION     VALUE 'ADD'.
    88  CHANGE-TRANSACTION  VALUE 'CHG'.
    88  POSTING-TRANSACTION VALUE 'PST'.
    88  VALID-UPDATE-CODE   VALUE 'ADD'
                                  'DEL'
                                  'CHG'
                                  'PST'.

    02  FILLER        PIC X(15).

01  SWITCHES.
    05  PROGRAM-ABEND-SWITCH  PIC X  VALUE 'N'.
        88  PROGRAM-ABEND            VALUE 'Y'.
    05  FILE-STATUS-FLAG            PIC XX VALUE '00'.
        88  SUCCESSFUL-COMPLETION          VALUE '00'.
        88  DUPLICATE-ALTERNATE-KEY        VALUE '02'.
        88  END-OF-FILE                    VALUE '10'.
        88  SEQUENCE-ERROR                 VALUE '21'.
        88  DUPLICATE-KEY                  VALUE '22'.
        88  NO-RECORD-FOUND                VALUE '23'.
        88  BOUNDARY-VIOLATION-IND-REL     VALUE '24'.
        88  PERMANENT-I-O-DATA-ERROR       VALUE '30'.
```

Figure JS1-8 Listing of Sample Indexed Sequential File Update Program for Inventory (Part 2 of 7)

```
          88  BOUNDARY-VIOLATION-SEQ          VALUE '34'.
          88  LOGIC-ERROR                     VALUE '92'.
       05  I-O-OPERATION-FLAG         PIC X(07) VALUE '       '.
   01  REPORT-TITLE.
       02  FILLER         PIC X(25) VALUE SPACES.
       02  TITLE          PIC X(45) VALUE
          'INVENTORY STATUS REPORT FOR JOB INC.,'.
       02  FILLER         PIC X(22) VALUE SPACES.
       02  FILLER         PIC X(4) VALUE 'PAGE'.
       02  FILLER         PIC X VALUE SPACES.
       02  PAGE-CNT-OUT   PIC Z9.
       02  FILLER         PIC X(10) VALUE SPACES.
   01  COLUMN-HEADINGS.
       02  FILLER         PIC X(5) VALUE SPACES.
       02  FILLER  PIC X(25) VALUE 'PART NO.   DESCRIPTION    '.
       02  FILLER         PIC X(19) VALUE SPACES.
       02  FILLER  PIC X(24) VALUE 'AMT ON HAND   PRICE EACH'.
       02  FILLER         PIC X(22) VALUE '       INVENTORY VALUE'.
       02  FILLER         PIC X VALUE SPACES.
       02  FILLER         PIC X(10) VALUE 'REODER-PT'.
       02  FILLER         PIC X VALUE SPACES.
       02  FILLER         PIC X(6) VALUE 'STATUS'.
       02  FILLER         PIC X(20) VALUE SPACES.
   01  DETAIL-LINE.
       02  FILLER         PIC X(6) VALUE SPACES.
       02  P-PART-NUM     PIC X(5).
       02  FILLER         PIC X(5) VALUE SPACES.
       02  P-DESCRIPTION  PIC X(30).
       02  FILLER         PIC X(6) VALUE SPACES.
       02  P-AMT-ON-HAND  PIC ZZZZ9.
       02  FILLER         PIC X(7) VALUE SPACES.
       02  P-PRICE-PER    PIC $$$$9.99.
       02  FILLER         PIC X(7) VALUE SPACES.
       02  P-INV-VAL      PIC $$$$,$$$,$$9.99.
       02  FILLER         PIC X(4) VALUE SPACES.
       02  REOPT-OUT      PIC ZZZZ9.
       02  FILLER         PIC X(4) VALUE SPACES.
       02  P-UPDATE-ACTION PIC X(25) VALUE SPACES.
   01  IO-ERROR-LINE REDEFINES DETAIL-LINE.
       05  FILLER                      PIC X(01).
       05  FILLER                      PIC X(01).
       05  IO-FILE-STATUS-FLAG         PIC X(02).
       05  FILLER                      PIC X(01).
       05  IO-OPERATION-FLAG           PIC X(07).
       05  FILLER                      PIC X(01).
       05  IO-ERROR-KEY                PIC X(05).
       05  FILLER                      PIC X(114).
*
*
```

```
 PROCEDURE DIVISION.
*
 000-MAIN-CONTROL-ROUTINE.
     PERFORM 100-INITIALIZATION.
     PERFORM 200-PROCESS-AND-READ UNTIL END-OF-JOB.
     PERFORM 400-CLOSING-ROUTINE.
     STOP RUN.
*
 100-INITIALIZATION.
     OPEN I-O   INVEN-OLD-MASTER-FILE
          INPUT INVEN-TRANSACTION-FILE.
     OPEN OUTPUT
         PRINT-FILE.
     PERFORM 800-READ-TRANSACTION-RECORD.
     PERFORM 700-NEW-PAGE-ROUTINE.
*
 200-PROCESS-AND-READ.
     MOVE SPACE TO DETAIL-LINE.
     IF VALID-UPDATE-CODE
         PERFORM 810-READ-MASTER-RECORD.
     IF (ADD-TRANSACTION AND NO-RECORD-FOUND)
         MOVE 'N' TO PROGRAM-ABEND-SWITCH
         PERFORM 230-ADD-MASTER-RECORD
     ELSE
         IF (CHANGE-TRANSACTION AND SUCCESSFUL-COMPLETION)
             PERFORM 240-CHANGE-MASTER-RECORD
          ELSE
             IF (POSTING-TRANSACTION AND SUCCESSFUL-COMPLETION)
                PERFORM 245-POSTING-ROUTINE
             ELSE
                IF (DELETE-TRANSACTION AND SUCCESSFUL-COMPLETION)
                    PERFORM 250-DELETE-MASTER-RECORD
                ELSE
                    PERFORM 290-IDENTIFY-ERROR-TYPE.
     PERFORM 720-PRINT-TRANSACTION.
     IF NOT PROGRAM-ABEND
        PERFORM 800-READ-TRANSACTION-RECORD
     ELSE
        PERFORM 900-ABORT-PROGRAM-RUN.
*
 230-ADD-MASTER-RECORD.
     PERFORM 260-FORMAT-MASTER-RECORD.
     PERFORM 850-WRITE-MASTER-RECORD.
     IF NOT PROGRAM-ABEND
         MOVE 'ADDED    ' TO P-UPDATE-ACTION.
*
 240-CHANGE-MASTER-RECORD.
     PERFORM 270-CHANGE-MASTER-FIELDS.
     PERFORM 855-REWRITE-MASTER-RECORD.
```

Figure JS1-8 Listing of Sample Indexed Sequential File Update Program for Inventory (Part 3 of 7)

```
     IF NOT PROGRAM-ABEND
         MOVE ' CHANGED ' TO P-UPDATE-ACTION.
*
 245-POSTING-ROUTINE.
     COMPUTE M-AMT-ON-HAND ROUNDED = M-AMT-ON-HAND +
         AMT-ON-HAND.
     PERFORM 855-REWRITE-MASTER-RECORD.

*****  THIS IS AN EXAMPLE OF A PROGRAM STUB
*
 250-DELETE-MASTER-RECORD.
     PERFORM 860-DELETE-MASTER-RECORD.
     IF NOT PROGRAM-ABEND
         MOVE 'DELETED ' TO P-UPDATE-ACTION.
*
 260-FORMAT-MASTER-RECORD.
      MOVE PART-NUM TO M-PART-NUM
      MOVE DESCRIPTION TO M-DESCRIPTION
      MOVE AMT-ON-HAND TO M-AMT-ON-HAND
      MOVE REORDER-PT TO M-REORDER-PT
      MOVE OPTIMUM-LVL TO M-OPTIMUM-LVL
      MOVE MAX-QUAN TO M-MAX-QUAN
      MOVE PRICE-PER TO M-PRICE-PER.
*
 270-CHANGE-MASTER-FIELDS.
     IF DESCRIPTION NOT EQUAL SPACES
         MOVE DESCRIPTION TO M-DESCRIPTION.
     IF AMT-ON-HAND NOT EQUAL SPACE
         MOVE AMT-ON-HAND TO M-AMT-ON-HAND.
     IF REORDER-PT NOT EQUAL SPACES
         MOVE REORDER-PT TO M-REORDER-PT.
     IF OPTIMUM-LVL NOT EQUAL TO SPACES
         MOVE OPTIMUM-LVL TO M-OPTIMUM-LVL.
     IF MAX-QUAN NOT EQUAL TO SPACES
         MOVE MAX-QUAN TO M-MAX-QUAN.
     IF PRICE-PER NOT EQUAL TO SPACES
        MOVE PRICE-PER TO M-PRICE-PER.
 290-IDENTIFY-ERROR-TYPE.
     IF  ADD-TRANSACTION
         MOVE 'REJECTED, ALREADY ON' TO P-UPDATE-ACTION.
*
*ERROR ROUTINE  STUB
*
 400-CLOSING-ROUTINE.
     DISPLAY 'END OF INVENTORY MASTER FILE UPDATE PROGRAM'.
     CLOSE INVEN-OLD-MASTER-FILE
         INVEN-TRANSACTION-FILE
         PRINT-FILE.
```

```
*
 700-NEW-PAGE-ROUTINE.
     ADD 1 TO PAGE-CNT.
     MOVE PAGE-CNT TO PAGE-CNT-OUT.
     WRITE PRINT-LINE FROM REPORT-TITLE
         AFTER ADVANCING TO-TOP-OF-PAGE.
     WRITE PRINT-LINE FROM COLUMN-HEADINGS
         AFTER ADVANCING 2 LINES.
     MOVE 0 TO LINE-CNT.
 720-PRINT-TRANSACTION.
     MOVE PART-NUM TO P-PART-NUM
     MOVE DESCRIPTION TO P-DESCRIPTION
     MOVE AMT-ON-HAND TO P-AMT-ON-HAND
     MOVE PRICE-PER TO P-PRICE-PER
     MOVE REORDER-PT TO REOPT-OUT.
     MOVE INV-VAL TO P-INV-VAL
     IF LINE-CNT GREATER THAN 29 PERFORM 700-NEW-PAGE-ROUTINE.
     WRITE PRINT-LINE FROM DETAIL-LINE
         AFTER ADVANCING 2 LINES.
     ADD 1 TO LINE-CNT.
 800-READ-TRANSACTION-RECORD.
     READ INVEN-TRANSACTION-FILE INTO WS-TRAN
         AT END DISPLAY 'ERR AT END OF TRANS FILE'.
     IF PART-NUM EQUAL '99999'
        MOVE 'Y' TO END-OF-JOB-SWITCH.
     DISPLAY 'READ-T = ' WS-TRAN.
 810-READ-MASTER-RECORD.
     MOVE PART-NUM TO IS-PART-NO.
     MOVE 'READ ' TO I-O-OPERATION-FLAG.
     READ INVEN-OLD-MASTER-FILE INTO WS-OLD-MAST
         INVALID KEY MOVE 'Y' TO PROGRAM-ABEND-SWITCH.
     IF SUCCESSFUL-COMPLETION
         NEXT SENTENCE
     ELSE
         MOVE 'Y' TO PROGRAM-ABEND-SWITCH.
     DISPLAY 'READ-M = ' WS-OLD-MAST.
     DISPLAY 'STATUS  = ' FILE-STATUS-FLAG
          ' OPERATION = ' I-O-OPERATION-FLAG.
 850-WRITE-MASTER-RECORD.
     MOVE M-PART-NUM TO IS-PART-NO.
     MOVE 'WRITE  ' TO I-O-OPERATION-FLAG.
     WRITE INVEN-OLD-MASTER-RECORD FROM WS-OLD-MAST
          INVALID KEY MOVE 'Y' TO PROGRAM-ABEND-SWITCH.
     IF SUCCESSFUL-COMPLETION
           NEXT SENTENCE
     ELSE
          MOVE 'Y' TO PROGRAM-ABEND-SWITCH.
     DISPLAY 'STATUS  = ' FILE-STATUS-FLAG
             ' OPERATION = ' I-O-OPERATION-FLAG.
```

Figure JS1-8 Listing of Sample Indexed Sequential File Update Program for Inventory (Part 4 of 7)

```
**
 855-REWRITE-MASTER-RECORD.
     MOVE  'REWRITE' TO I-O-OPERATION-FLAG.
     MOVE M-PART-NUM TO IS-PART-NO.
     REWRITE INVEN-OLD-MASTER-RECORD
        FROM WS-OLD-MAST
        INVALID KEY MOVE 'Y' TO PROGRAM-ABEND-SWITCH.
     IF SUCCESSFUL-COMPLETION
         NEXT SENTENCE
     ELSE
         MOVE 'Y' TO PROGRAM-ABEND-SWITCH.
     DISPLAY 'STATUS  = ' FILE-STATUS-FLAG
             ' OPERATION = ' I-O-OPERATION-FLAG.
*
 860-DELETE-MASTER-RECORD.
     MOVE 'DELETE ' TO I-O-OPERATION-FLAG.
     MOVE M-PART-NUM TO IS-PART-NO.
     DELETE INVEN-OLD-MASTER-FILE
         INVALID KEY MOVE 'Y' TO PROGRAM-ABEND-SWITCH.
     IF SUCCESSFUL-COMPLETION
         NEXT SENTENCE
     ELSE
         MOVE 'Y' TO PROGRAM-ABEND-SWITCH.
     DISPLAY 'STATUS  = ' FILE-STATUS-FLAG
             ' OPERATION = ' I-O-OPERATION-FLAG.
```

```
*
 900-ABORT-PROGRAM-RUN.
     MOVE FILE-STATUS-FLAG TO IO-FILE-STATUS-FLAG.
     MOVE I-O-OPERATION-FLAG TO IO-OPERATION-FLAG.
     MOVE IS-PART-NO TO IO-ERROR-KEY.
     MOVE 'ABORTED ' TO P-UPDATE-ACTION
     DISPLAY 'PROGRAM ABORT, IO-ERRER-LINE =' IO-ERROR-LINE.
     MOVE 'Y' TO END-OF-JOB-SWITCH.
```

Figure JS1-8 Listing of Sample Indexed Sequential File Update Program for Inventory (Part 5 of 7)

```
READ-T = I19010TORSION BAR                   22000                     CHG       ISI04940
READ-M = I19010TORSION BAR                   25000300004000050000000035ADD
STATUS  = 00 OPERATION = READ
STATUS  = 00 OPERATION = REWRITE
READ-T = I19030                                                        DEL       ISI04950
READ-M = I19030STRIKER                       60000700008000090000000002ADD
STATUS  = 00 OPERATION = READ
STATUS  = 00 OPERATION = DELETE
READ-T = I19045EXTENSION SPRING              99999444126512375000000010CHG       ISI04960
READ-M = I19045EXTENSION SPRING              99900444006500075000000010ADD
STATUS  = 00 OPERATION = READ
STATUS  = 00 OPERATION = REWRITE
READ-T = I19045                              00112                     PST       ISI04970
READ-M = I19045EXTENSION SPRING              99999444126512375000000010ADD
STATUS  = 00 OPERATION = READ
STATUS  = 00 OPERATION = REWRITE
READ-T = I20005                                                        DEL       ISI04980
READ-M = I20005COMPRESSION SPRING            07517350004000050000000100ADD
STATUS  = 00 OPERATION = READ
STATUS  = 00 OPERATION = DELETE
READ-T = I20005COMPRESSION SPRING-POST 83 MOD07500350004000050000000110ADD       ISI04990
READ-M = I20005COMPRESSION SPRING            07517350004000050000000100ADD
STATUS  = 23 OPERATION = READ
STATUS  = 00 OPERATION = WRITE
READ-T = I20005                              07517                     CHG       ISI05000
READ-M = I20005COMPRESSION SPRING-POST 83 MOD07500350004000050000000110ADD
STATUS  = 00 OPERATION = READ
STATUS  = 00 OPERATION = REWRITE
READ-T = I20005                              00017                     PST       ISI05010
READ-M = I20005COMPRESSION SPRING-POST 83 MOD07517350004000050000000110ADD
STATUS  = 00 OPERATION = READ
STATUS  = 00 OPERATION = REWRITE
READ-T = I2012012AMP BREAKER                 19400250004100055000000350ADD       ISI05020
READ-M = I2012012AMP BREAKER                 19400250004100055000000350ADD
STATUS  = 00 OPERATION = READ
READ-T = I20145LAG FUSE                      20400180002500035000000016ADD       ISI05030
READ-M = I20145LAG FUSE                      20400180002500035000000016ADD
STATUS  = 00 OPERATION = READ
READ-T = I20165                                                        DEL       ISI05040
READ-M = I20165HEX NUT                       17800150002500035000000005ADD
STATUS  = 00 OPERATION = READ
STATUS  = 00 OPERATION = DELETE
READ-T = I20170OUTLET                        22505250004000050000000069CHG       ISI05050
READ-M = I20170OUTLET                        22505250004000050000000069ADD
STATUS  = 00 OPERATION = READ
STATUS  = 00 OPERATION = REWRITE
READ-T = I20180SWITCH PLATE                  20500180005000075000000020CHG       ISI05060
READ-M = I20180SWITCH PLATE                  20500180005000075000000020ADD
STATUS  = 00 OPERATION = READ
STATUS  = 00 OPERATION = REWRITE
READ-T = I20200LABELS                        18175140003000042000000001ADD       ISI05070
READ-M = I20200LABELS                        18175140003000042000000001ADD
STATUS  = 00 OPERATION = READ
READ-T = I20225CONTACTORS                    18844220004000062000000010ADD       ISI05080
READ-M = I20225CONTACTORS                    18844220004000062000000010ADD
STATUS  = 00 OPERATION = READ
READ-T = I2023520 VOLT POWER ASSY            19500180003000050000000050ADD       ISI05090
READ-M = I2023520 VOLT POWER ASSY            19500180003000050000000050ADD
STATUS  = 00 OPERATION = READ
```

Figure JS1-8 Listing of Sample Indexed Sequential File Update Program for Inventory (Part 6 of 7)

```
READ-T = I20240                                                  DEL       ISI05100
READ-M = I20240FRICTION LATCH         19100500006000090000000025ADD
STATUS  = 00 OPERATION = READ
STATUS  = 00 OPERATION = DELETE
READ-T = I20240FRICTION LATCH MOD A   00013400006000090000000027ADD       ISI05110
READ-M = I20240FRICTION LATCH         19100500006000090000000025ADD
STATUS  = 23 OPERATION = READ
STATUS  = 00 OPERATION = WRITE
READ-T = I20240FRICTION LATCH MODEL A                            CHG       ISI05120
READ-M = I20240FRICTION LATCH MOD A   00013400006000090000000027ADD
STATUS  = 00 OPERATION = READ
STATUS  = 00 OPERATION = REWRITE
READ-T = I20240                       000234                     PST       ISI05130
READ-M = I20240FRICTION LATCH MODEL A 00013400006000090000000027ADD
STATUS  = 00 OPERATION = READ
STATUS  = 00 OPERATION = REWRITE
READ-T = I20276TERMINAL BLOCK SUB-ASSEMBLY   17000100001200015000000050ADD       ISI05140
READ-M = I20240FRICTION LATCH MODEL A 00036400006000090000000027ADD
STATUS  = 23 OPERATION = READ
STATUS  = 00 OPERATION = WRITE
READ-T = I99999TRAILER FOR INVENTORY TRANS   0000000000000000000000000           ISI05150
END OF INVENTORY MASTER FILE UPDATE PROGRAM
```

Figure JS1-8 Listing of Sample Indexed Sequential File Update Program for Inventory (Part 7 of 7)

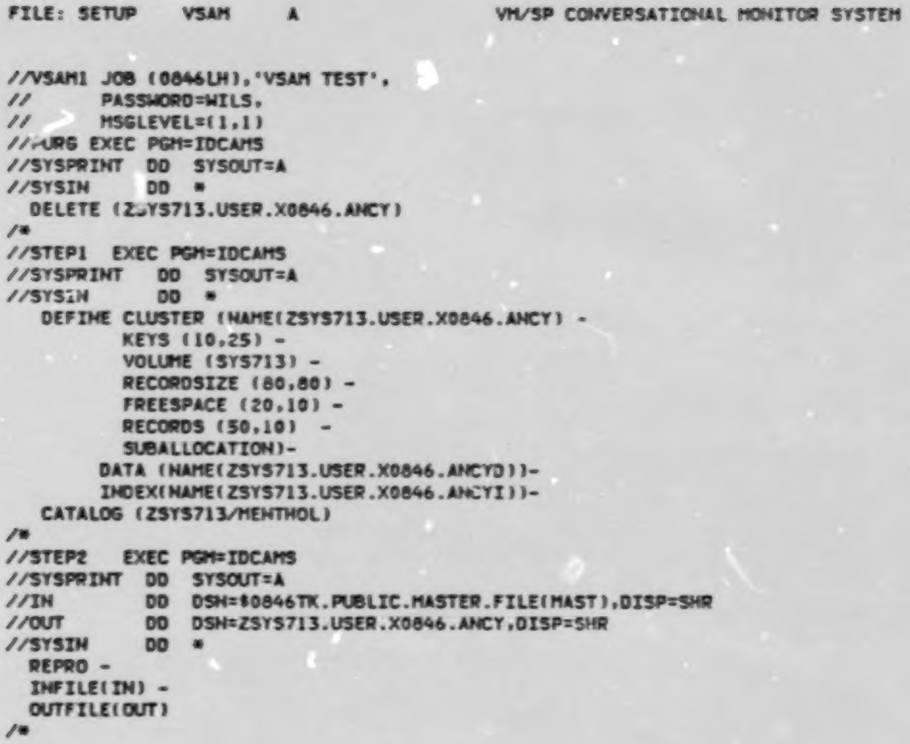
```
                    INVENTORY STATUS REPORT FOR JOB INC.,                     PAGE  1
PART NO.   DESCRIPTION                   AMT ON HAND   PRICE EACH       INVENTORY VALUE REODER-PT  STATUS
 19010     TORSION BAR                      22000        $0.00                $0.00        0      CHANGED
 19030                                          0        $0.00                $0.00        0     DELETED
 19045     EXTENSION SPRING                 99999        $0.10                $0.00    44412      CHANGED
 19045                                        112        $0.00                $0.00        0
 20005                                          0        $0.00                $0.00        0     DELETED
 20005     COMPRESSION SPRING-POST 83 MOD    7500        $1.10                $0.00    35000     ADDED
 20005                                       7517        $0.00                $0.00        0      CHANGED
 20005                                         17        $0.00                $0.00        0
 20120     12AMP BREAKER                    19400        $3.50                $0.00    25000     REJECTED, ALREADY ON
 20145     LAG FUSE                         2040)        $0.16                $0.00    18000     REJECTED, ALREADY ON
 20165                                          0        $0.00                $0.00        0     DELETED
 20170     OUTLET                           22505        $0.69                $0.00    25000      CHANGED
 20180     SWITCH PLATE                     20500        $0.20                $0.00    18000      CHANGED
 20200     LABELS                           18175        $0.01                $0.00    14000     REJECTED, ALREADY ON
 20225     CONTACTORS                       18844        $0.10                $0.00    22000     REJECTED, ALREADY ON
 20235     20 VOLT POWER ASSY               19500        $0.50                $0.00    18000     REJECTED, ALREADY ON
 20240                                          0        $0.00                $0.00        0     DELETED
 20240     FRICTION LATCH MOD A                13        $0.27                $0.00    40000     ADDED
 20240     FRICTION LATCH MODEL A               0        $0.00                $0.00        0      CHANGED
 20240                                         23        $0.00                $0.00    40000
 20276     TERMINAL BLOCK SUB-ASSEMBLY      17000        $0.50                $0.00    10000     ADDED
```

Program H1 - Setup Solution for Payroll (non VSAM portion) (Part 1 of 2)

```
FILE: FILESUP  CNTL     A   UNIVERSITY OF MISSOURI COMPUTER NETWORK

//FILES JOB (1136LH),'JOHN MAS',
//       PASSWORD=
//       MSGLEVEL=(1,1)
//PURG EXEC PGM=IEHPROGM
//SYSPRINT  DD  SYSOUT=A
//DD2       DD UNIT=3350,VOL=SER=SYS710,DISP=OLD
//SYSIN     DD  *
  SCRATCH DSNAME=$1136TK.PUBLIC.MASTER.FILE,VOL=3350=SYS710
  UNCATLG DSNAME=$1136TK.PUBLIC.MASTER.FILE
/*
//COPY2    EXEC PGM=IEBGENER
//SYSPRINT DD SYSOUT=*
//SYSIN    DD DUMMY
//SYSUT2   DD DSN=$1136TK.PUBLIC.MASTER.FILE(DATA),
//            DISP=(NEW,CATLG,DELETE),
//            DCB=(RECFM=FB,LRECL=80,BLKSIZE=80,DSORG=PO),
//            UNIT=3350,VOL=SER=SYS710,
//            SPACE=(TRK,(2,1,1),RLSE)
//SYSUT1   DD *
BURKE, SADIE            BUR-145927DATA PROCESSING039501580000126400
BURDEN, JACK            BUR-382196ACCOUNTING     073502940000220500
BUTLER, RHETT           BUT-824571MARKETING      087503500000262500
FAVIR, LARRY            FAV-835127MARKETING      070002800000210000
FINKBINE, GEORGE        FIN-571433ACCOUNTING     047501900000150000
HAMILTON, MELANIE       HAM-446735DATA PROCESSING038501732500130600
HAMPTON, WADE           HAM-642151DATA PROCESSING043201728000138240
IRWIN, MONTAGUE         IRW-924586ACCOUNTING     085603424000256800
MACINTOSH, ANGUS        MAC-662473MARKETING      053002226000178080
MASTERN, CASS           MAS-619318MARKETING      045001912500153000
MERRIWETHER, MAYBELLE   MER-527348DATA PROCESSING027501100000093500
MORRISON, JAMES         MOR-754884DATA PROCESSING057301942000156700
OHARA,SCARLETT          OHA-431552MARKETING      045001800000144000
PORTNOY, ALEXANDER      POR-783914DATA PROCESSING100004000000300000
ROBILLARD, ELLEN        ROB-445331DATA PROCESSING036001440000115200
SLATTERY, THOMAS        SLA-108423ACCOUNTING     093503740000280500
STANTON, ANNE           STA-274432DATA PROCESSING0905037400C0280500
STARK, WILLIE           STA-367810MARKETING      060202528000192240
TARLETON, BRENT         TAR-311834DATA PROCESSING037501500000120000
WATLING, BELLE          WAT-623489MARKETING      063202528000189600
WYNDER, ABLE            WYN-246372ACCOUNTING     075003000000240000
TRAILER FOR MASTER      ZZZ-999999                   4911900379966021
/*
//
```

BEST COPY AVAILABLE

Program H1 - Setup Solution for Payroll
(VSAM portion) (Part 2 of 2)



```
FILE: SETUP    VSAM     A                 VM/SP CONVERSATIONAL MONITOR SYSTEM

//VSAM1 JOB (0846LH),'VSAM TEST',
//      PASSWORD=WILS,
//      MSGLEVEL=(1,1)
//PURG EXEC PGM=IDCAMS
//SYSPRINT  DD  SYSOUT=A
//SYSIN     DD  *
  DELETE (ZSYS713.USER.X0846.ANCY)
/*
//STEP1  EXEC PGM=IDCAMS
//SYSPRINT   DD  SYSOUT=A
//SYSIN      DD  *
   DEFINE CLUSTER (NAME(ZSYS713.USER.X0846.ANCY) -
          KEYS (10,25) -
          VOLUME (SYS713) -
          RECORDSIZE (80,80) -
          FREESPACE (20,10) -
          RECORDS (50,10)  -
          SUBALLOCATION)-
        DATA (NAME(ZSYS713.USER.X0846.ANCYD))-
        INDEX(NAME(ZSYS713.USER.X0846.ANCYI))-
   CATALOG (ZSYS713/MENTHOL)
/*
//STEP2   EXEC PGM=IDCAMS
//SYSPRINT  DD  SYSOUT=A
//IN        DD  DSN=$0846TK.PUBLIC.MASTER.FILE(MAST),DISP=SHR
//OUT       DD  DSN=ZSYS713.USER.X0846.ANCY,DISP=SHR
//SYSIN     DD  *
  REPRO -
  INFILE(IN) -
  OUTFILE(OUT)
/*
```

2. Program H2 - Indexed Sequential Solution for Payroll (Procedure Division) (Part 1 of 3)

```
 PROCEDURE DIVISION.
*
 000-MAIN-CONTROL-ROUTINE.
    PERFORM 100-INITIALIZATION.
    PERFORM 200-READ-AND-PROCESS
       UNTIL END-OF-JOB.
    PERFORM 400-CLOSING-ROUTINE.
    STOP RUN.
*
 100-INITIALIZATION.
    OPEN I-O  OLD-MASTER-FILE
        INPUT TRANSACTION-FILE.
    OPEN OUTPUT
        PRINT-FILE.
    PERFORM 800-READ-TRANSACTION-RECORD.
    PERFORM 910-NEW-PAGE-ROUTINE.
*
 200-READ-AND-PROCESS.
    MOVE SPACES TO DETAIL-LINE.
    IF VALID-UPDATE-CODE
       PERFORM 810-READ-MASTER-RECORD.
    IF (ADDITION AND NO-RECORD-FOUND)
       MOVE 'N' TO PROGRAM-ABEND-SWITCH
       PERFORM 230-ADD-MASTER-RECORD
    ELSE
       IF (CHANGE AND SUCCESSFUL-COMPLETION)
          PERFORM 240-CHANGE-MASTER-RECORD
       ELSE
          IF (POSTING AND SUCCESSFUL-COMPLETION)
             PERFORM 245-POSTING-ROUTINE
          ELSE
             IF (DELETION AND SUCCESSFUL-COMPLETION)
                PERFORM 250-DELETE-MASTER-RECORD
             ELSE
                PERFORM 290-IDENTIFY-ERROR-TYPE.
    PERFORM 750-PRINT-TRANSACTION.
    IF NOT PROGRAM-ABEND
       PERFORM 800-READ-TRANSACTION-RECORD
    ELSE
       PERFORM 900-ABORT-PROGRAM-RUN.
*
 230-ADD-MASTER-RECORD.
    PERFORM 260-FORMAT-MASTER-RECORD.
    PERFORM 850-WRITE-MASTER-RECORD.
    IF NOT PROGRAM-ABEND
       MOVE ' ADDED ' TO DL-UPDATE-ACTION.
 240-CHANGE-MASTER-RECORD.
    PERFORM 270-CHANGE-MASTER-FIELDS.
    PERFORM 855-REWRITE-MASTER-RECORD.
    IF NOT PROGRAM-ABEND
       MOVE ' CHANGED ' TO DL-UPDATE-ACTION.
*
 245-POSTING-ROUTINE.
    IF GROSS-PAY IS NUMERIC
       ADD GROSS-PAY TO TOTAL-GROSS-PAY
       ADD GROSS-PAY TO OLD-GROSS-PAY-YTD.
    IF NET-PAY IS NUMERIC
       ADD NET-PAY TO TOTAL-NET-PAY
       ADD NET-PAY TO OLD-NET-PAY-YTD.
    PERFORM 855-REWRITE-MASTER-RECORD.
*
 250-DELETE-MASTER-RECORD.
    PERFORM 860-DELETE-MASTER-RECORD.
       IF NOT PROGRAM-ABEND
          MOVE ' DELETED ' TO DL-UPDATE-ACTION.
 260-FORMAT-MASTER-RECORD.
    MOVE EMPLOYEE-NAME TO OLD-EMPLOYEE-NAME.
    MOVE EMPLOYEE-ID TO OLD-EMPLOYEE-ID.
    MOVE DEPARTMENT TO OLD-DEPARTMENT.
    MOVE PAY-RATE TO OLD-PAY-RATE.
    MOVE GROSS-PAY TO OLD-GROSS-PAY-YTD.
    MOVE NET-PAY TO OLD-NET-PAY-YTD.
```

BEST COPY AVAILABLE

BEST COPY AVAILABLE

2. Program H2 - Indexed Sequential Solution for Payroll (Procedure Division) (Part 2 of 3)

```
*
 270-CHANGE-MASTER-FIELDS.
     IF EMPLOYEE-NAME NOT EQUAL SPACES
         MOVE EMPLOYEE-NAME TO OLD-EMPLOYEE-NAME.
     IF EMPLOYEE-ID NOT EQUAL SPACES
         MOVE EMPLOYEE-ID TO OLD-EMPLOYEE-ID.
     IF DEPARTMENT NOT EQUAL SPACES
         MOVE DEPARTMENT TO OLD-DEPARTMENT.
     IF PAY-RATE NOT EQUAL SPACES
         MOVE PAY-RATE TO OLD-PAY-RATE.
     IF GROSS-PAY NOT EQUAL SPACES
         MOVE GROSS-PAY TO OLD-GROSS-PAY-YTD.
     IF NET-PAY NOT EQUAL SPACES
         MOVE NET-PAY TO OLD-NET-PAY-YTD.
*
 290-IDENTIFY-ERROR-TYPE.
     IF ADDITION
         MOVE 'REJECTED, ALREADY ON' TO DL-UPDATE-ACTION
     ELSE
         IF CHANGE
             MOVE 'MASTER NOT FOUND' TO DL-UPDATE-ACTION
         ELSE
             IF POSTING
                 MOVE 'MASTER NOT FOUND' TO DL-UPDATE-ACTION
             ELSE
                 IF DELETION
                     MOVE 'INVALID TRANS. TYPE'
                         TO DL-UPDATE-ACTION.
*
 400-CLOSING-ROUTINE.
     DISPLAY 'END OF PAYROLL MASTER FILE UPDATE PROGRAM'.
     CLOSE OLD-MASTER-FILE
           TRANSACTION-FILE
           PRINT-FILE.
*
 750-PRINT-TRANSACTION.
     MOVE TRANSACTION-TYPE TO DL-TRANSACTION-TYPE
     MOVE EMPLOYEE-ID TO DL-EMPLOYEE-ID
     MOVE EMPLOYEE-NAME TO DL-EMPLOYEE-NAME
     MOVE DEPARTMENT TO DL-DEPARTMENT
     MOVE PAY-RATE TO DL-PAY-RATE
     MOVE GROSS-PAY TO DL-GROSS-PAY
     MOVE NET-PAY TO DL-NET-PAY
     IF LINE-COUNT GREATER THAN LINE-LIMIT
         PERFORM 910-NEW-PAGE-ROUTINE.
     WRITE PRINT-LINE FROM DETAIL-LINE
         AFTER ADVANCING 2 LINES.
     ADD 2 TO LINE-COUNT.
*
 800-READ-TRANSACTION-RECORD.
     READ TRANSACTION-FILE INTO WS-TRANSACTION-RECORD
         AT END DISPLAY 'ERR AT END OF TRANS FILE'.
     IF EMPLOYEE-ID EQUAL 'ZZZ-999999'
         MOVE 'Y' TO END-OF-JOB-SWITCH.
     DISPLAY 'READ-T = ' WS-TRANSACTION-RECORD.
*
 810-READ-MASTER-RECORD.
     MOVE EMPLOYEE-ID TO KEY-EMPLOYEE-ID.
     MOVE 'READ ' TO I-O-OPERATION-FLAG.
     READ OLD-MASTER-FILE INTO WS-OLD-MASTER-RECORD
         INVALID KEY MOVE 'Y' TO PROGRAM-ABEND-SWITCH.
     IF SUCCESSFUL-COMPLETION
         NEXT SENTENCE
     ELSE
         MOVE 'Y' TO PROGRAM-ABEND-SWITCH.
     DISPLAY 'READ-M = ' WS-OLD-MASTER-RECORD.
     DISPLAY 'STATUS = ' FILE-STATUS-FLAG
             'OPERATION = ' I-O-OPERATION-FLAG.
```

2. Program H2 - Indexed Sequential Solution for Payroll (Procedure Division) (Part 3 of 3)

```
*
 850-WRITE-MASTER-RECORD.
     MOVE EMPLOYEE-ID TO KEY-EMPLOYEE-ID.
     MOVE 'WRITE ' TO I-O-OPERATION-FLAG.
     WRITE OLD-MASTER-RECORD FROM WS-OLD-MASTER-RECORD
         INVALID KEY MOVE 'Y' TO PROGRAM-ABEND-SWITCH.
     IF SUCCESSFUL-COMPLETION
         NEXT SENTENCE
     ELSE
         MOVE 'Y' TO PROGRAM-ABEND-SWITCH.
     DISPLAY 'STATUS = ' FILE-STATUS-FLAG
             'OPERATION = ' I-O-OPERATION-FLAG
             'READ-M = ' WS-OLD-MASTER-RECORD.
*
 855-REWRITE-MASTER-RECORD.
     MOVE 'REWRITE' TO I-O-OPERATION-FLAG.
     MOVE OLD-EMPLOYEE-ID TO KEY-EMPLOYEE-ID.
     REWRITE OLD-MASTER-RECORD
         FROM WS-OLD-MASTER-RECORD
         INVALID KEY MOVE 'Y' TO PROGRAM-ABEND-SWITCH.
     IF SUCCESSFUL-COMPLETION
         NEXT SENTENCE
     ELSE
         MOVE 'Y' TO PROGRAM-ABEND-SWITCH.
     DISPLAY 'STATUS = ' FILE-STATUS-FLAG
             'OPERATION = ' I-O-OPERATION-FLAG
             'READ-M = ' WS-OLD-MASTER-RECORD.
*
 860-DELETE-MASTER-RECORD.
     MOVE 'DELETE' TO I-O-OPERATION-FLAG.
     MOVE EMPLOYEE-ID TO KEY-EMPLOYEE-ID.
     DELETE OLD-MASTER-FILE
         INVALID KEY MOVE 'Y' TO PROGRAM-ABEND-SWITCH.
     IF SUCCESSFUL-COMPLETION
         NEXT SENTENCE
     ELSE
         MOVE 'Y' TO PROGRAM-ABEND-SWITCH.
     DISPLAY 'STATUS = ' FILE-STATUS-FLAG
             'OPERATION = ' I-O-OPERATION-FLAG
             'READ-M = ' WS-OLD-MASTER-RECORD.
*
 900-ABORT-PROGRAM-RUN.
     MOVE FILE-STATUS-FLAG TO IO-FILE-STATUS-FLAG.
     MOVE I-O-OPERATION-FLAG TO IO-OPERATION-FLAG.
     MOVE KEY-EMPLOYEE-ID TO IO-ERROR-KEY.
     MOVE 'ABORTED ' TO DL-UPDATE-ACTION
     DISPLAY 'PROGRAM ABORT, IO-ERROR-LINE =' IO-ERROR-LINE.
     MOVE 'Y' TO END-OF-JOB-SWITCH.
*
 910-NEW-PAGE-ROUTINE.
     WRITE PRINT-LINE FROM REPORT-TITLE
         AFTER ADVANCING TO-TOP-OF-PAGE.
     WRITE PRINT-LINE FROM HEADING-LINE-1
         AFTER ADVANCING 2 LINES.
     WRITE PRINT-LINE FROM HEADING-LINE-2
         AFTER ADVANCING 1 LINES.
     MOVE 8 TO LINE-COUNT.
*
```

BEST COPY AVAILABLE

BEST COPY AVAILABLE

3. Program H3 (Changed routines and printouts) (Part 1 of 2)

```
*
*
 INPUT-OUTPUT SECTION.
 FILE-CONTROL.
     SELECT INVEN-TRANSACTION-FILE ASSIGN TO UR-S-SYSIN.
     SELECT INVEN-OLD-MASTER-FILE
         ASSIGN TO DA-UMCSR1
         ORGANIZATION IS INDEXED
         ACCESS MODE IS DYNAMIC
         RECORD KEY IS F-PART-NUM
         FILE STATUS IS WS-FILE-STATUS-FLAG.
     SELECT PRINT-FILE ASSIGN TO UR-S-SYSPRINT.

 PROCEDURE DIVISION.
 000-MAIN-CONTROL-RTN.
     PERFORM 850-MASTER-DISPLAY.
     PERFORM 010-INITIALIZATION.
     PERFORM 300-INTERACTIVE-READ-PROCESS.
     PERFORM 200-READ-PROCESS UNTIL MORE-RECORDS EQUAL 'N'.
     PERFORM 900-CLOSING.
     STOP RUN.
*
 850-MASTER-DISPLAY.
      OPEN INPUT INVEN-OLD-MASTER-FILE.
      IF MORE-RECORDS EQUAL 'Y'
          DISPLAY '            ***BEGINNING MASTER FILE***'
      ELSE
          DISPLAY '            ***UPDATED MASTER FILE***'.
      MOVE 'Y' TO MORE-RECORDS.
      PERFORM 855-READ-AND-DISPLAY UNTIL MORE-RECORDS EQUAL 'N'.
      DISPLAY '            ***END OF MASTER FILE***'.
      CLOSE INVEN-OLD-MASTER-FILE.
*
 855-READ-AND-DISPLAY.
      READ INVEN-OLD-MASTER-FILE NEXT RECORD INTO WS-OLD-MAST
      AT END MOVE 'N' TO MORE-RECORDS.
      DISPLAY ' MASTER:  ' WS-OLD-MAST.
*
 900-CLOSING.
     CLOSE INVEN-TRANSACTION-FILE,
         PRINT-FILE,
         INVEN-OLD-MASTER-FILE.
      PERFORM 850-MASTER-DISPLAY.
*
```

576

BEST COPY AVAILABLE

3. Program H3 (Changed routines and printouts) (Part 2 of 2)

Before updating:

```
BURKE, SADIE              BUR-145927DATA PROCESSING039501580000126400
BURDEN, JACK              BUR-382196ACCOUNTING     073502940000220500
BUTLER, RHETT             BUT-824571MARKETING      087503500000262500
FAVIR, LARRY              FAV-835127MARKETING      070002800000210000
FINKBINE, GEORGE          FIN-571433ACCOUNTING     047501900000150000
HAMILTON, MELANIE         HAM-446735DATA PROCESSING038501732500138600
HAMPTON, WADE             HAM-642151DATA PROCESSING043201728000138240
IRWIN, MONTAGUE           IRW-924586ACCOUNTING     085603424000256800
MACINTOSH, ANGUS          MAC-662473MARKETING      053002226000178080
MASTERN, CASS             MAS-619318MARKETING      045001912500153000
MERRIWETHER, MAYBELLE     MER-527348DATA PROCESSING027501100000093500
MORRISON, JAMES           MOR-754884DATA PROCESSING057301942000156700
OHARA SCARLETT            OHA-431552MARKETING      045001800000144000
PORTNOY, ALEXANDER        POR-783914DATA PROCESSING100004000000300000
ROBILLARD, ELLEN          ROB-445331DATA PROCESSING036001440000115200
SLATTERY, THOMAS          SLA-108423ACCOUNTING     093503740000280500
STANTON, ANNE             STA-274432DATA PROCESSING090503740000280500
STARK, WILLIE             STA-367810MARKETING      060202528000202240
TARLETON, BRENT           TAR-311834DATA PROCESSING037501500000120000
WATLING, BELLE            WAT-623489MARKETING      063202528000189600
WYNDER, ABLE              WYN-246372ACCOUNTING     075003000000240000
                          ZZZ-999999                   4911900379966020
```

After updating:

```
BURKE, SADIE              BUR-145927DATA PROCESSING039501738000139040
BURDEN, JOHN              BUR-382196ACCOUNTING     073502940000220500
BUTLER, RHETT             BUT-824571MARKETING      087503500000262500
CALVERT, CATHLEEN         CAL-336741MARKETING      06250     00     0
FINKBINE, GEORGE          FIN-571433ACCOUNTING     047502090000165200
HAMILTON, MELANIE         HAM-446735DATA PROCESSING038501732500138600
HAMPTON, WADE             HAM-642151DATA PROCESSING043201728000138240
MACINTOSH, ANGUS          MAC-662473MARKETING      053002226000176080
MASTERN, CASS             MAS-619318MARKETING      050002112500169000
MERRIWETHER, MAYBELLE     MER-527348DATA PROCESSING027501100000093500
MORRISON, JAMES           MOR-754884DATA PROCESSING057301942000156700
OHARA,SCARLETT            OHA-431552MARKETING      045001800000144000
PORTNOY, ALEXANDER        POR-783914DATA PROCESSING100004000000300000
ROBILLARD, ELLEN          ROB-445331DATA PROCESSING049901440000115200
SLATTERY, THOMAS          SLA-108423ACCOUNTING     093504114000308550
STANTON, ANNE             STA-274432DATA PROCESSING090504140500310775
STARK, WILLIE             STA-367810MARKETING      060202528000202240
SUTTER, BRUCE             SUT-654321DATA PROCESSING15000     00     0
TARLETON, BRENT           TAR-311834DATA PROCESSING037501500000120000
WATLING, BELLE            WAT-623489MARKETING      063202528000189600
WILKES, ASHLEY            WIL-300177ACCOUNTING     08350     00     0
WYNDER, ABLE              WYN-246372ACCOUNTING     075003000000240000
ZIPPER, ANN               ZIP-714000CLERICAL       04500000000000000020
                          ZZZ-999999                   4911900379966020
```

INDEXED SEQUENTIAL FILES II

UNIT XIII

EXAMINATION

1. Match the definitions in Column A with the corresponding phrase in Column B.

	Column A		Column B
______a.	pairs of keys and address	1.	index
______b.	random update	2.	direct update
		3.	necessary for random update of COBOL files
______c.	OLRT	4.	on line real time
______d.	record key	5.	used to find a particular record
______e.	OPEN I-O		

2. Why isn't there any batching, sorting, or holding of transactions in random updates?

TRUE AND FALSE

T F 1. A record is deleted by not rewriting it to the master file.

T F 2. A record which is changed will be rewritten back to the master file.

4. Write a valid COBOL sentence to rewrite a record called "MASTER-RECORD" and use the invalid key clause to turn on switch called "AT-END-SW".

5. In the program listing what is the purpose of the data item?

01 NOMINAL-KEY-FIELD.

INDEXED SEQUENTIAL FILES II

UNIT XIII

ANSWERS TO EXAMINATION

1. __1__ a
   __2__ b
   __4__ c
   __5__ d
   __3__ e

2. Because these are replaced with a terminal user inputting the transactions.

3. a. F
   b. T

4. REWRITE MASTER-RECORD INVALID KEY MOVE 'Y' TO AB-END-SW.

5. Its value is set to the value of the key of the record which is to be retrieved, thus it is used to specify which master record is obtained from the master file.

BUSINESS DATA PROCESSING IN BASIC

UNIT XIV

UNIT OBJECTIVE

After studying this unit, the student should be able to write file-oriented, small business data processing application programs in BASIC for microcomputers.

SPECIFIC OBJECTIVES

After studying this unit, the student should be able to:

1. Match terms and definitions.
2. Recognize correct statements concerning the role of BASIC in small business data processing.
3. Explain characteristics of microcomputers which are oriented to small business data processing applications.
4. Write file-oriented BASIC commands.
5. Recognize correct statements concerning the design of the accounts receivable sample application.
6. Identify functions of details in the computer systems design for the sample application.
7. Explain the function of the modules in the design of a program.
8. List steps in processing of a program by reading the documentation.
9. Write a print program to complete a sample application of microcomputers in a small business.

BUSINESS DATA PROCESSING IN BASIC

UNIT XIV

SUGGESTED ACTIVITIES

I. Instructor

   A. Make transparencies; prepare lectures, using this curriculum guide and supplemental materials.

   B. Provide students with objectives; discuss objectives.

   C. Make parallel assignments for reading and programming from the text(s).

   D. Provide students with information and assignment sheets; discuss these sheets.

   E. Provide students with additional supplementary information and exercises.

   F. Give examination.

II. Student

   A. Read objectives.

   B. Study information sheets and test assignments.

   C. Complete assignment sheets and supplementary exercises.

   D. Complete examination.

INSTRUCTIONAL MATERIALS

I. Unit content

   A. Objectives

   B. Notes to the instructor

   C. Information sheets

   D. Handouts

      1. HO 1: Pseudocode for Main Menu Program, "RMENU.BAS"

      2. HO 2: Source Listing for Main Menu Program, "RMENU.BAS"

      3. HO 3: Pseudocode for "RNEWCUS.BAS" Program to Add New Customers to Accounts Receivable File for Rogers Company

    4. HO 4: Source Listing for the New Customer Program, "RNEWCUS.BAS"
    5. HO 5: Source Listing for the Sales and Payments Program, "RSALPAY.BAS"
    6. HO 6: Pseudocode for "RPRINT.BAS", Program to Print Accounts Receivable

E. Transparency masters

    1. TM 1: Typical Application of Microcomputers
    2. TM 2: An Example Menu
    3. TM 3: The Rogers Company Accounts Receivable System
    4. TM 4: Sample Design of the Main Menu Screen
    5. TM 5: Design of the Screen for Addition of New Customers
    6. TM 6: Hierarchy Chart for the New Customer Program, "RNEWCUS.BAS"
    7. TM 7: Design of the Sales and Payments Screen
    8. TM 8: Hierarchy Chart for the Sales and Payments Program
    9. TM 9: Overview of the Print Statements Program

E. Assignment sheets

    1. AS 1: BASIC File Commands
    2. AS 2: BASIC Buffer Commands

G. Answers to assignment sheets

H. Job sheet 1: Print Program for Rogers Company Accounts Receivable

I. Examination

J. Answers to examination

BUSINESS DATA PROCESSING IN BASIC

UNIT XIV

NOTES TO THE INSTRUCTOR

1. It is important to key and execute the sample solution programs before the course starts because BASIC is not very standard.

2. Provide a copy of the sample system on floppy disk for each student or team of students.

3. Most problems in debugging will occur with inconsistent movements of data into and out of the buffers. Check your computer manual for the correct method on your system.

4. More emphasis was placed on relative files rather than sequential files because of their usage in small business data processing and for ease of support of direct file access.

5. All solutions and samples were done on the IBM PC microcomputer.

BUSINESS DATA PROCESSING IN BASIC

UNIT XIV

INFORMATION SHEETS

I. Terms and functional definitions

A. **Relative (file organization)** - type of file in which records are identified by their position relative to the beginning of the file

B. **Buffer** - the input/output and record definition area for a BASIC file (different from other languages)

C. **OPEN** - command to prepare a file for processing

D. **CLOSE** - command to end processing of a file

E. **PUT** - command to take a record from its buffer and to write that record to a file on external media, such as a diskette

F. **LSET** - command to left justify data

G. **FIELD** - command to define a record and its associated buffer

H. **GET** - command to fetch a record from a file on external media and place it into the buffer

I. **VAL(M$)** - function to return the numeric value of a string variable, M$

J. **RUN** - command to cause the starting of a program

K. **SYSTEM** - command to return control to the operating system

L. **Accounts Receivable System - balance forward** - simple accounting system which records amounts owed by customers

M. **Menu** - display on the CRT screen of choices for the user

N. **Prompts** - requests to the user for information or action, usually displayed on the CRT screen

O. **Visual verification** - control technique whereby user checks data on screen

II. Introduction to data processing on microcomputers

A. The relationship of programming in BASIC to careers in data processing

1. BASIC is now the most widely used language in the world because of its popularity on microcomputers.

2. Professional opportunities exist for the data processing person who can develop business applications for small companies or small offices of large companies.

3. Applications generally involve a complete solution to a small business' accounting or recordkeeping needs, such as:

   a. Accounts receivable/billing

   b. Inventory

   c. Payroll

4. This unit will explain and develop a small but complete accounts receivable system which would be typical of credit sales in a retail setting.

B. Knowledge requirements

1. It is assumed that the student has completed a course in introductory BASIC.

2. This unit covers only files and file-oriented commands which are necessary for small business applications. Other BASIC knowledge is assumed.

III. Overview of interactive data processing applications on microcomputers

A. BASIC Files

1. A file in BASIC resides on external media such as magnetic tape, cassette, floppy disk, or hard disk.

2. A file is composed of records, which in turn are composed of fields (which are called variables). Fields are related by being in the same record. The record consists of these fields.

3. In business applications, a file is processed one record at a time. One record is brought into main memory and then processed. In other BASIC applications, an entire small file might be brought into memory in the form of a table; the entire table is then processed. This tabular processing is seldom used in business data processing and will not be covered.

4. Most business applications on microcomputers are interactive in nature, so most files reside on floppy disks to enable random access. Random access on floppy disks will be assumed.

B. Characteristics of microcomputer-oriented, small business data processing applications

1. TM 1 is a schematic flowchart of a typical application of a microcomputer for small business data processing.

(NOTE: In TM 1 the user is at the computer entering the transaction at the keyboard and receiving messages on the CRT. This is an important design characteristic; all programs should converse with the user and be interactive in a user-friendly fashion.

C. Characteristics of interactive and user-friendly applications

1. Menus - The user is presented with a set of simplified choices. By entering the number of the choice desired, the system automatically loads and starts the program(s) to accomplish the desired function, such as to enter the receipt of merchandise into the inventory. (TM 2)

2. Prompting - At every step in the processing, the program tells or asks the user what to do next.

   (NOTE: The line at the bottom of the CRT screen in TM 2 is an example of a prompt.)

D. Division into separate modules or programs

1. As in COBOL, the primary design strategy is to break the application design into small modules and to develop these small modules in an independent fashion through pseudocode, IPO charts, flowcharts, etc.

2. The modules are brought together into programs through hierarchy charts.

3. Unlike COBOL, the programs are generally kept very small, less than 200 lines, and are restricted to a single function. For example, in TM 2, each menu choice would generally be a separate program. Thus, the menus also help in planning the required set of programs.

4. Small, single-function programs are an important aid to decreasing maintenance costs.

IV. BASIC's file-oriented commands

A. File creation, opening, and closing

1. Files in BASIC are created by the system as soon as an OPEN statement is encountered. The file will be empty, but it will exist.

2. All files must be opened before they are used.

3. Format of the OPEN statement is:

   OPEN "R",1,"file name"

   a. In this format, the R stands for Relative, the form of direct access in which the record key is relative to the beginning of the file, e.g., record key "1" is the first record; record key "2" is the second record, etc.

   b. The "1" indicates an identification number, which is to be assigned to the input/output or buffer area for this file. Later statements will use this number to access the buffer for this file.

c. The file name further identifies the file for control purposes. A typical file name is "RMENU.BAS".

B. Closing files

1. All files must be closed before a program finishes.

2. Format

CLOSE, which closes all files

or

CLOSE file-number, which closes the specific file and leaves the others open.

C. Using files and records

1. In BASIC, records are PUT into files and later the GET command brings a record back into memory from the external file. A record's data must be set into the buffer before the PUT and taken from the buffer after the GET statement.

2. Before a record-oriented command can be issued to the file, a FIELD command must exist in the program to define a record and its associated buffer.

(NOTE: This is like a DIM statement to define a matrix.)

3. Format of the FIELD statement is:

```
FIELD buffer number, length-1 AS name-1, length-2 AS name-2, . . .
```

The buffer number cross references the definition of the record back to the file in which the record is to be used. The length is best explained by example. For instance, if you wanted to define a record composed of a name and address, the statement might be:

```
FIELD 1, 20 AS NAME, 25 AS ADDRESS
```

Here the first 20 characters are reserved for name and the next 25 characters for addresses.

4. Numeric items in files must be translated from numeric mode into string mode and left shifted before being PUT out into the file. This is accomplished with the LSET command. For example:

```
LSET A6$ = STR $(D5)
```

Simultaneously translates the contents of D5 (a numeric variable) into string mode, and sets it into A6$ in left adjusted mode. This is a source of confusion, but if careful attention is paid to examples of PUT's and their related LSET's, the principle will be clear.

5. Format of the PUT statement.

   PUT buffer no. record key

   In this format the buffer number again cross references the record back to its definition and the record key (usually taken from the transaction) tells which relative master record is to be written (PUT) into the file.

6. Obtaining records from relative files is done through the GET statement.

7. Format of the GET statement:

   GET buffer no. record key

   Here, buffer number again cross-references the record to the definition in the field statement and record key tells which relative record number is to be retrieved from the file.

8. Numeric variables which are obtained from the files must be converted back to numeric (but not right shifted) before being used. For example:

   AMT = VAL (A6$)

   Converts A6$'s value back into numeric from string and places it in the variable called AMT. AMT should not be in the buffer, i.e., named in the FIELD statement.

D. Running programs and returning

   1. Programs can call upon the operating system of the microcomputer to run another program. For example, the menu program typically calls upon another program to print the file contents. This is done through the RUN command.

   2. Format of the RUN command:

      RUN "program name"

      In this format the program name is the name which had been assigned to the program by SAVEing it. This command is used by a program (or the user) to cause execution of another program.

   3. To return control to the operating system the command is simply:

      SYSTEM

      This command would be used by the menu program to end processing by the application.

V. Overview of the sample system

A. Accounts receivable for the Rogers Company

1. Accounts receivable systems keep records on what is owed to the company and by whom. Generally the file is organized by customer number as the key.

2. Balance forward is the simplest accounts receivable system; it contains in each record the name, address, and amount owed by each customer. No attempt is made to record specific items purchased by the customers.

B. The Rogers Company's Accounts Receivable System:

1. The accounts receivable system for the Rogers Company is depicted in TM 3.

2. Concerning TM 3:

a. The user/salesperson will approach the system having just made a credit sale with a sales slip in hand. The customer's information will be entered from the sales slip on the keyboard by the salesperson.

b. The main menu, point A, will ask the salesperson what option s/he would like to use:

1. Add a new customer

2. Enter a sale to an existing customer or credit a payment from an existing customer to their account

3. Print statements for all the customers (monthly activity)

4. Quit accounts receivable processing

(NOTE: Option 3 would normally be done at the end of the month.)

c. Each of the options causes another program to be run, which in turn displays its menu and prompts on the CRT. Also, options 1 and 2 cause the affected records in the accounts receivable master file to be changed or updated.

d. Option 3 prints the statements for billing of the customers. This should not normally be done by a salesperson, but in a small company there may be no other personnel.

e. There will be four programs in the system: main menu, new customers, sales and payments, and print statements.

VI. Design of the sample system - approach and main menu program

A. Approach to the design

1. Portions of the system will be made available as a guide to the student. The student will complete the system by writing one of the four programs.

2. Pseudocode for each program will be given to assist the student.

B. Overview and pseudocode for the main menu program

1. An overview of the design of the main program's menu (TM 4)

2. Pseudocode for the main menu program (HO 1)

a. The program's design is quite simple. It paints the menu, accepts a choice or indicates an error, and then runs the chosen program.

C. The listing for the main menu program (HO 2)

1. The CLS command clears the screen to remove any previous writing.

2. The REM commands are remarks for documentation.

3. The print statement paints the menu.

4. The variable, CH, contains the choice of which program to run.

5. The RUN statement causes the chosen program to be loaded from disk into memory and to begin executing.

VII. Design of the sample system - the new customer program

A. Overview of the design of the screen for new customers (TM 5)

1. The screen is a series of prompts rather than a complete menu.

2. The initialize file prompt is necessary to set records in the file to zero. Initialization is done when the program is first run or when a new file is being built.

3. Hierarchy chart for the new customer program (TM 6)

1. The design of this program is to be accomplished in four modules.

2. The four modules are:

a. Main - prints main menu screen, controls input, and controls other modules

b. Initialization - sets ID's on the file to zero to initialize them. (Print program can then tell which ID's are not used and can avoid billing unused ID's.)

c. Error check - checks that ID is between 0 and 20, the allowable range of ID's

d. Output - left justifies numeric fields, converts numeric to string, and writes the record on the file

C. Pseudocode for the new customer program (HO 3)

1. In HO 3, at line 20 the user is prompted for the routine to initialize the file.

2. At line 50, the new customer information is obtained.

3. At line 70, the output routine is performed.

4. At line 80, the program asks if there are any more new customers. If not, it closes and returns to the main menu program.

D. Source listing for the new customer program (HO 4)

VIII. Design of the sample system - the sales and payment program, "RSALPAY.BAS"

A. An overview of the combined sales and payment program's prompts and screen (TM 7)

1. First, the program asks for customer number so that it can retrieve the customer's master record from the file.

2. The program then prints the name and current balance from the master file on the screen and asks for a visual verification that the correct account is being updated.

3. Next, the program asks whether it is a sale or payment, computes the new balance, and then displays the new balance before writing the new balance to the file.

4. The program finally asks if there are any more sales or payments; if not, it returns to the main menu program.

B. Hierarchy chart for the sales and payments program (TM 8)

1. In TM 8 the modules are:

a. Main - controls screen, gets new information, computes balance, controls other modules

b. Read - fetches the current master record

c. Write - adjusts data for output, writes updated master file

C. Source listing for the sales and payments program (HO 5)

1. At line 60 in HO 5, the program opens the master file.

2. At line 100, the program obtains the ID of the customer and then goes to the read module to obtain the customer's data.

3. At lines 120-130, the customer name and balance are displayed, and a visual verification is obtained from the user.

4. At 160-190, the amount of the sale or the payment is obtained, a new balance is calculated, and then the new balance is displayed.

5. At line 200, the write module is performed to write the updated record.

6. At lines 210-260, the program finds out if there is more data to be processed; if not, it returns to the main program.

IX. Design of the sample system - the print statements program

A. A flow diagram overview of the print statements program (TM 9)

1. This program is to sequentially print each of the records on the master file in billing format.

2. The file will be opened for random access, but a FOR loop will access each record from beginning to end; i.e., number one to number nineteen.

3. If a particular account is unused (key = zero), the program is not to print a statement for that account.

B. Pseudocode for print program (HO 6)

1. In HO 6, at line 20, open the file for direct access as before.

2. At line 30, the record key will be indexed in a FOR loop to sequentially obtain each record.

3. At line 70, the print run is done; note that unused records are not to be printed.

4. At line 80, I is incremented for the next pass back through the FOR loop starting again at line 30.

5. At line 90, close and return to the main menu.

C. Source listing for print program - to be provided by student. (See JS I for instructions.)

HO 1

Pseudocode for Main Menu Program, "RMENU.BAS"

```
 10 Paint menu on screen
 20 Input choice
 30 If choice is "1" goto 80
 40 If choice is "2" goto 90
 50 If choice is "3" goto 100
 60 If choice is "4" goto 110
 70 Print error message and start over
 80 Run add new customer program
 90 Run sales or payments program
100 Run print statements program
110 Return to system to quit
120 End program
```

HO 2

Source Listing for Main Menu Program, "RMENU.BAS"

```
5 CLS
10 REM PROGRAM NAME IS RMENU.BAS
20 REM PROGRAM STUB FOR MAIN MENU FOR ROGERS CO
30 PRINT "MAIN MENU FOR ACCOUNTS RECEIVABLE "
40 PRINT "1  IS FOR NEW CUSTOMERS"
50 PRINT "2  IS FOR SALES OF PAYMENTS"
60 PRINT "3  IS FOR PRINTING INVOICES"
70 PRINT "4  IS FOR QUITING"
80 INPUT "PLEASE ENTER YOUR CHOICE ";CH
90 IF CH <1 OR CH > 4 GOTO 30
100 ON CH GOTO  110, 120, 130, 140
110 RUN "RNEWCUS.BAS"
120 RUN "RSALPAY.BAS"
130 RUN "RPRINT.BAS"
140 SYSTEM
150 END
```

HO 3

Pseudocode for "RNEWCUS.BAS" Program to Add New Customers to Accounts Receivable File for Rogers Company

```
  5 REM's Prog Name, Author
 10 Clear
 20 If Initialization Desired, Do Initialization Rtn.
 30 Open File for Direct Access
 35 CLS
 40 Display Screen for Adding a Customer
 50 Input Cus ID, Name, Address: Address 2,
    Address 3, Amt.
 60 Error Check Rtn
 65 IF ERR GO TO 35
 70 Do Output Rtn
 80 Finished? No Goto 35
 82 Close
 86 Run "RMENU.BAS"
 90 REM Output Rtn
100 Convert ID & Amt to String
110 Field the Buffer Area
120 Lset Data Into Fields
130 Put #, ID
140 Return
150 REM Error Rtn
160 Check ID<1 or > 19?
170 Set Err Sw
180 Return
190 Init. Rtn
200 For 1=1 to 19
210 Set ID=0
220 Put #, ID
240 Next I
250 End
```

HO 4

Source Listing for the New Customer Program, "RNEWCUS.BAS"

```
10 REM PROGRAM NAME IS RNEWCUS.BAS
20 REM PROGRAM IS THE NEW CUSTOMER ADDITION RTN FOR ROGERS CO.
30 OPEN "R",1,"RAR.DAT"
40 ER =0: CLS
50 INPUT "DO YOU WANT TO INITILIZE THE FILE";YN$
60 IF YN$ = "Y" THEN GOSUB 410
70 INPUT " CUS ID = <dd>";ID
80 INPUT "LAST NAME, FIRST NAME MI. =";D1$
90 INPUT "ADDRESS LINE ONE IS..."; D2$
100 INPUT "ADDRESS LINE TWO IS     "; D3$
110 INPUT "ADDRESS LINE 3 IS...";D4$
120 INPUT "CURRENT BALENCE OR ZERO ";D5
130 GOSUB 370 ' ERROR RTN
140 IF ER =  1  THEN PRINT "INPUT ERROR"ELSE GOTO 160
150 GOTO 70
160 GOSUB 260 ' OUTPUT RTN
170 PRINT "LAST CUS NOW ADDED TO FILE CUS NO = ";ID
180 INPUT "DO YOU WANT TO ADD ANOTHER";YN$
190 IF YN$ = "Y" THEN YN$ ="N":GOTO 70
200 CLOSE
210 INPUT "RETURN TO MAIN MENU";YN$
220 IF YN$<> "Y" GOTO 40
230 PRINT "END ADD PROGRAM, LOADING RMENU TO RETURN TO MAIN MENU"
240 RUN "RMENU.BAS"
250 REM NORMAL END OF PROGRAM AND RETURN TO MAIN PROGRAM
260 REM OUTPUT RTN
270 FIELD 1, 2 AS A1$, 30 AS A2$,30 AS A3$, 30 AS A4$, 20 AS A5$, 7 AS A6$
280 LSET A1$=MKI$(ID)
285 LSET A2$= D1$
290 LSET A3$=D2$
300 LSET A4$=D3$
310 LSET A5$=D4$
320 LSET A6$= STR$(D5)
330 PUT 1,ID
340 PRINT "JUST PUTTED";ID
350 REM END OF OUTPUT TRN*************
360 RETURN
370 REM ERROR RTN
380 IF ID <1 OR ID >19 THEN ER =1 ELSE ER =0
390 RETURN
400 END
410 REM    *********** INITIALIZATION ROUTINE *******
420 REM  SETS ID TO ZERO
430 FOR I =1 TO 19
440 FIELD 1, 2 AS B1$,30 AS B2$,30 AS B3$, 30 AS B4$,20 AS B5$, 7 AS B6$
450 S1 = 0
460 LSET B1$= MKI$(S1)
470 PUT 1,I
480 NEXT I
490 RETURN
```

HO 5

Source Listing for the Sales and Payments Program, "RSALPAY.BAS"

```
10 REM program to
20 REM PROGRAM TO APPLY SALES OR PAYMENTS TO RAR.DAT OF  ROGERS CO CASE
30 REM
40 REM  PROGRAM NAME IS RSALPAY.BAS
50 CLS
60 OPEN "R",1,"RAR.DAT"
70 PRINT "          ******** SALES OR PAYMENTS **********"
80 PRINT: PRINT: PRINT
90 PRINT " THIS PROGRAM ENTERS EITHER SALES OR PAYMENTS"
100 INPUT "PLEASE ENTER CUSTOMER NUMBER(DD)";ID
110 GOSUB 270   'READ ROUTINE
120 PRINT "CUSTOMER NAME.....";A2$
130 PRINT "CURRENT BALENCE IS "; BAL
132 YN$=" "
140 INPUT "IS THIS CORRECT ";YN$
150 IF YN$= "N" GOTO 100
160 INPUT "IS THIS A SALE OR CREDIT( S OR C)";SC$
170 INPUT "PLEASE ENTER AMOUNT";AMT
180 IF SC$ = "S" THEN BAL = BAL + AMT ELSE BAL = BAL - AMT
190 PRINT "NEW BALENCE IS NOW ";BAL
200 GOSUB 320 'WRITE ROUTINE
210 YN$=" "
220 INPUT "DO YOU HAVE ANOTHER ";YN$
230 IF YN$= "Y" GOTO 100
240 CLOSE
250 PRINT "RETURNING TO MAIN MENU"
260 RUN "RMENU.BAS"
270 REM  ********** READ ROUTINE *******"
280 FIELD 1, 2 AS A1$, 30 AS A2$, 30 AS A3$, 30 AS A4$, 20 AS A5$, 7 AS A6$
290 GET 1, ID
300 BAL= VAL (A6$)
310 RETURN
320 REM ********** WRITE UPDATED RECORD ROUTINE **********
330 FIELD 1, 2 AS B1$, 30 AS B2$, 30 AS B3$, 30 AS B4$, 20 AS B5$, 7 AS B6$
340 LSET B1$= MKI$(ID)
350 LSET B2$=A2$
360 LSET B3$= A3$
370 LSET B4$ = A4$
380 LSET B5$= B5$
390 LSET B6$= STR$(BAL)
400 PUT 1, ID
410 RETURN
```

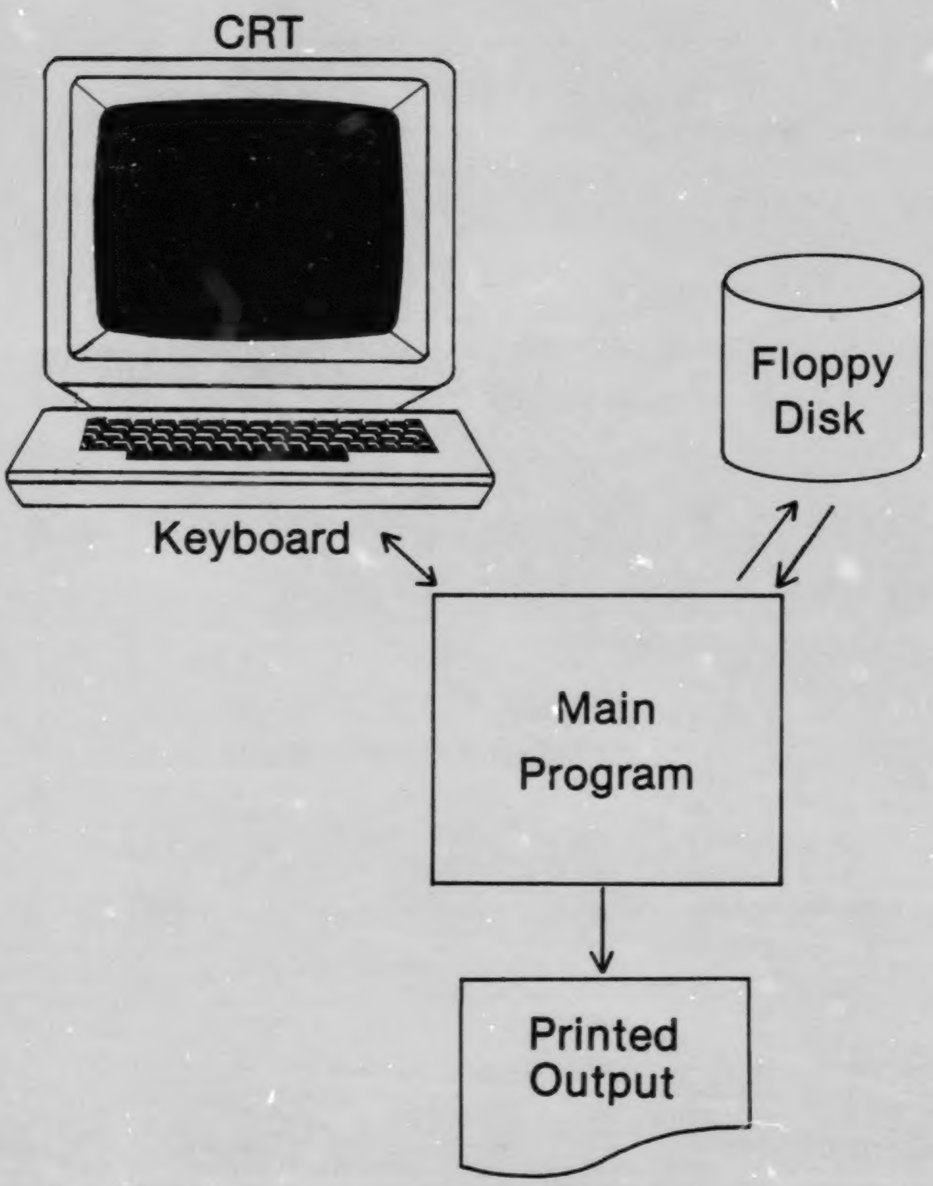
HO 6

Pseudocode for "RPRINT.BAS", Program to Print Accounts Receivable

```
 10 Remarks to Identify Program and Author
 20 Open the File for Direct Access as Before in
    RNEWCUS
 30 For I = 1 to Maximum No of Records Possible
    (twenty suggested)
 40 Field as in Other Program
 50 Get 1, I
    (Note this causes a quasi-sequential read of
    the random file)
 60 Convert ID and Amount to Numeric
    (Note use MKI$ for ID and Val for Amount)
 70 Gosub Print Routine
 80 Next I
 90 Close and Return to RMENU.BAS
100 REM    * Print Routine *
110 If it is not an Actively Used Record Don't Print It;
    Hint See the Initialization Routine in
    RNEWCUS.BAS
120 Print each Record as a Simple Statement (Your
    Instructor Will Tell You Whether to Print on
    the CRT or the Printer or Both)
130 Return
```

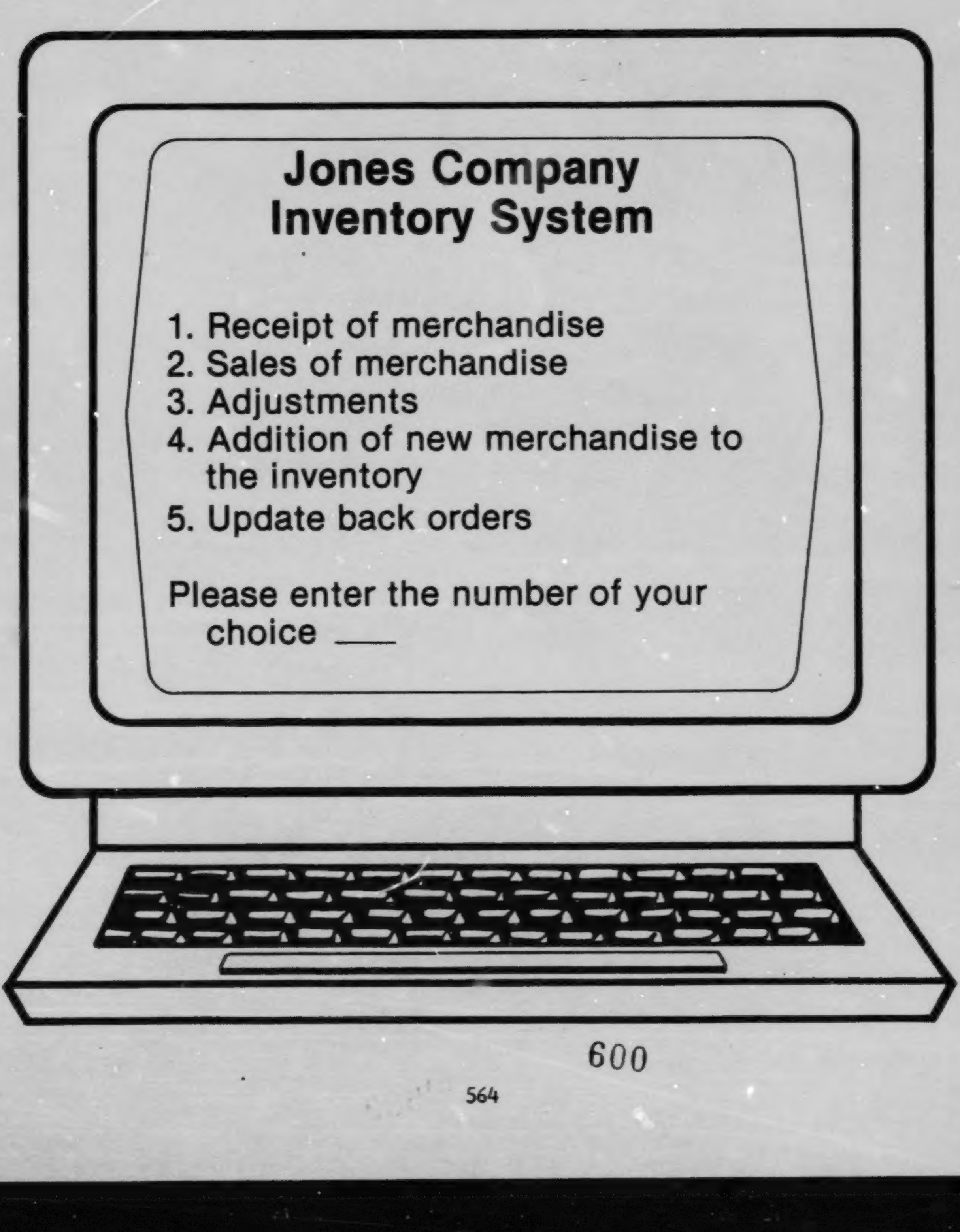
TM 1

Typical Application of Microcomputers


CRT
Floppy Disk
Keyboard
Main Program
Printed Output

TM 2

An Example MENU

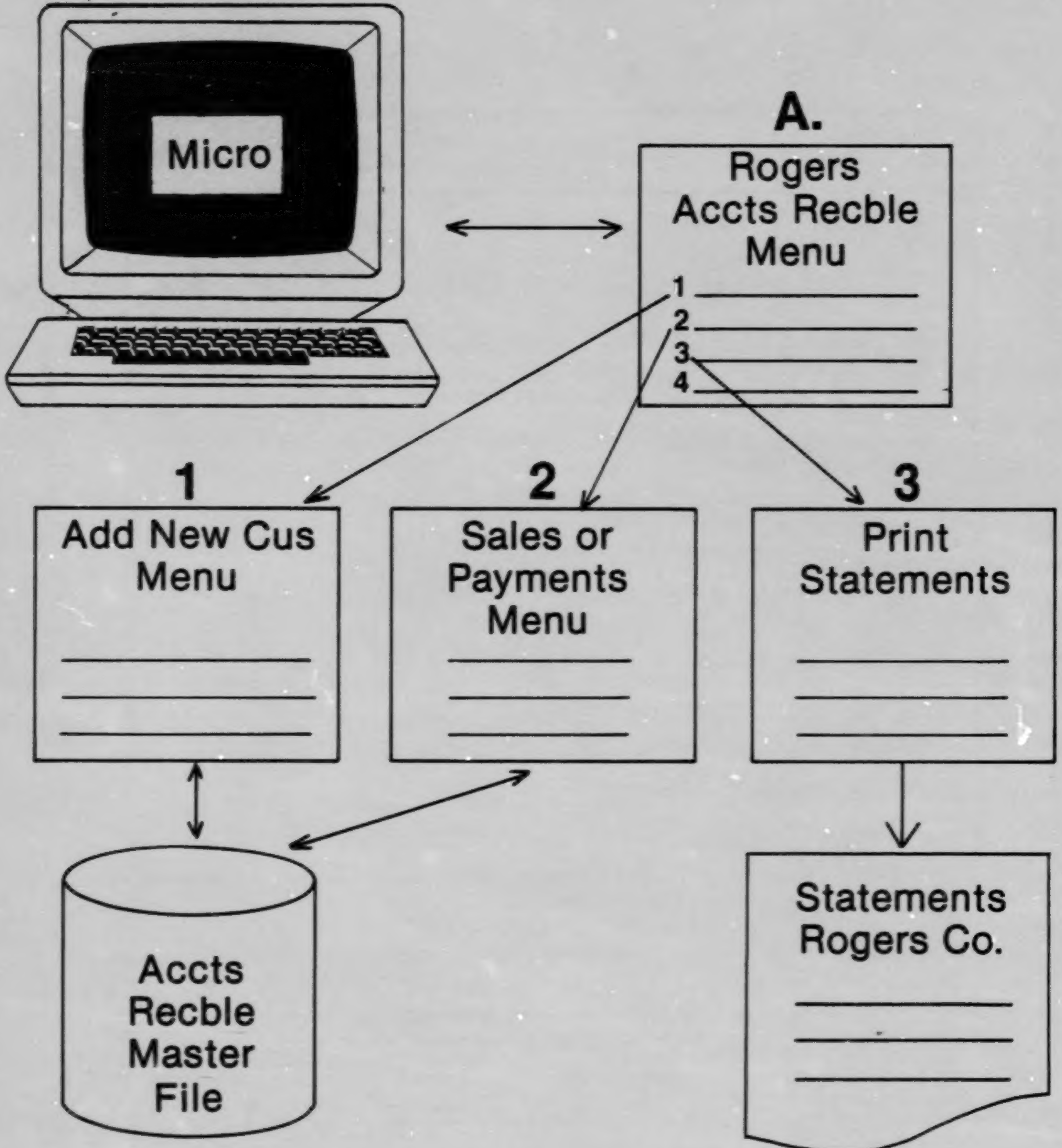


Jones Company
Inventory System

1. Receipt of merchandise
2. Sales of merchandise
3. Adjustments
4. Addition of new merchandise to the inventory
5. Update back orders

Please enter the number of your choice ___

The Rogers Company Accounts Receivable System



Micro

A.

Rogers Accts Recble Menu

1

2

3

4

1

Add New Cus Menu

2

Sales or Payments Menu

3

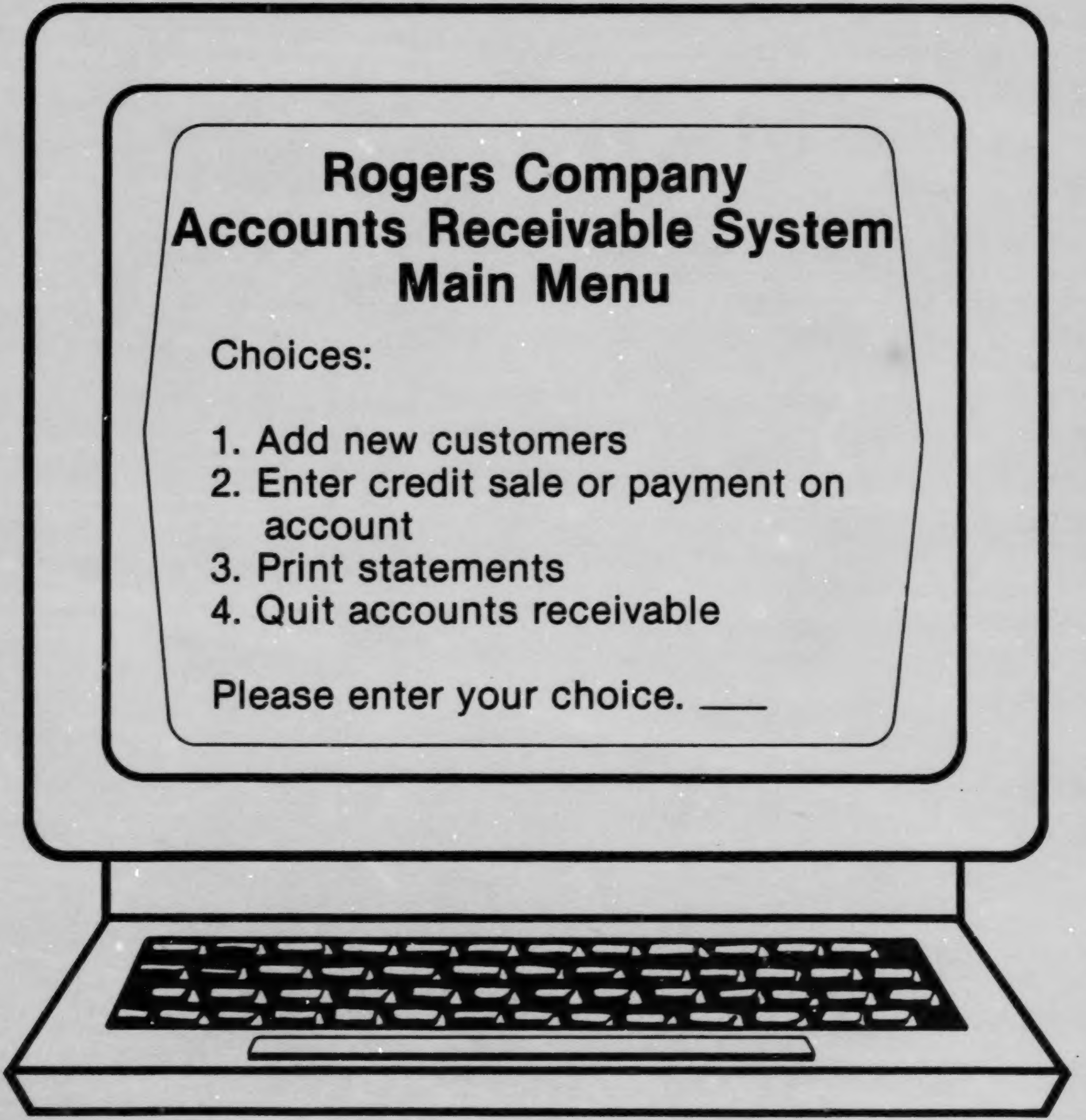
Print Statements

Accts Recble Master File

Statements Rogers Co.

TM 4

Sample Design of the Main Menu Screen



Rogers Company
Accounts Receivable System
Main Menu

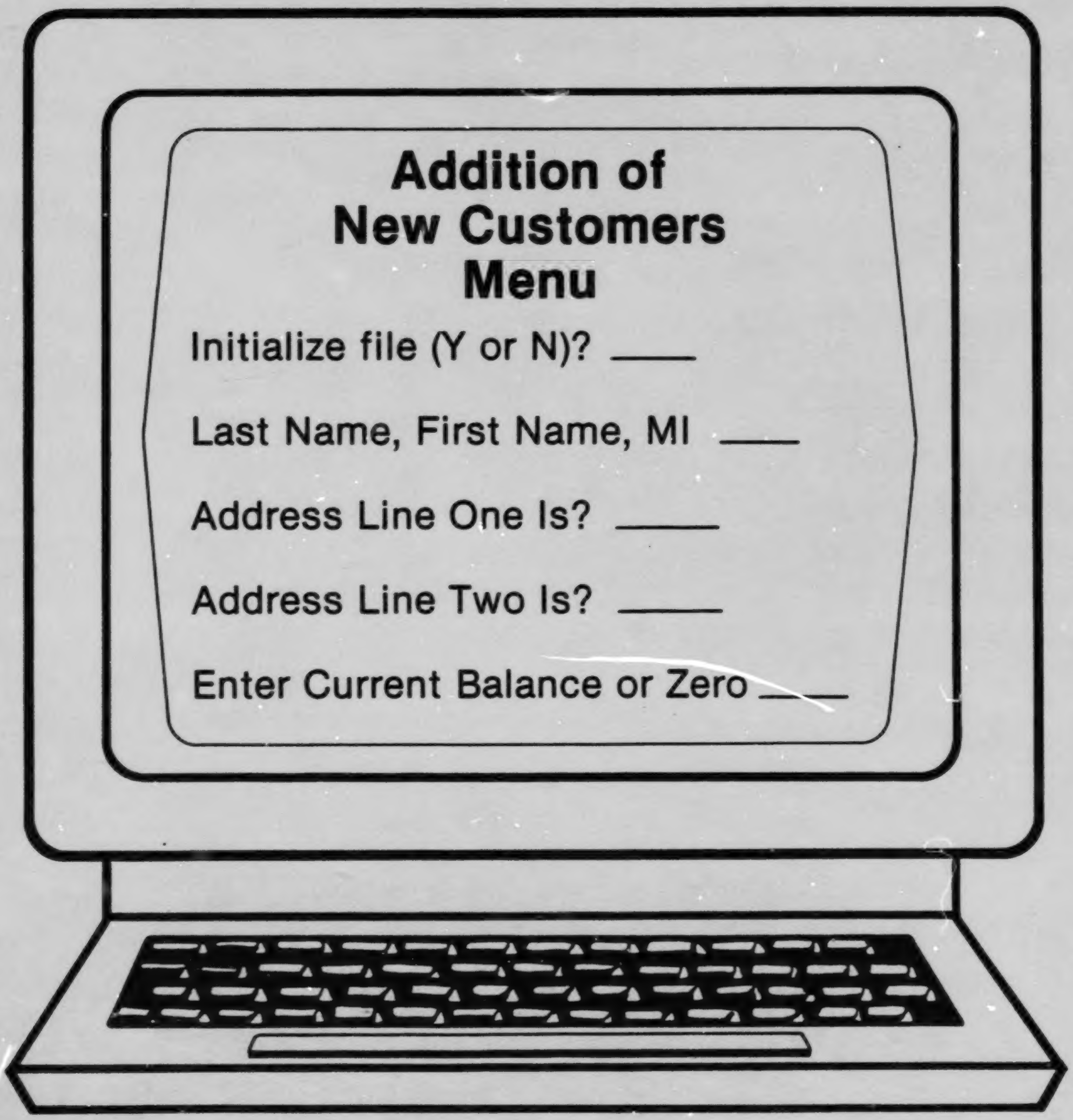
Choices:

1. Add new customers
2. Enter credit sale or payment on account
3. Print statements
4. Quit accounts receivable

Please enter your choice. ___

TM 5

Design of the Screen for Addition of New Customers



Addition of New Customers Menu

Initialize file (Y or N)? ____

Last Name, First Name, MI ____

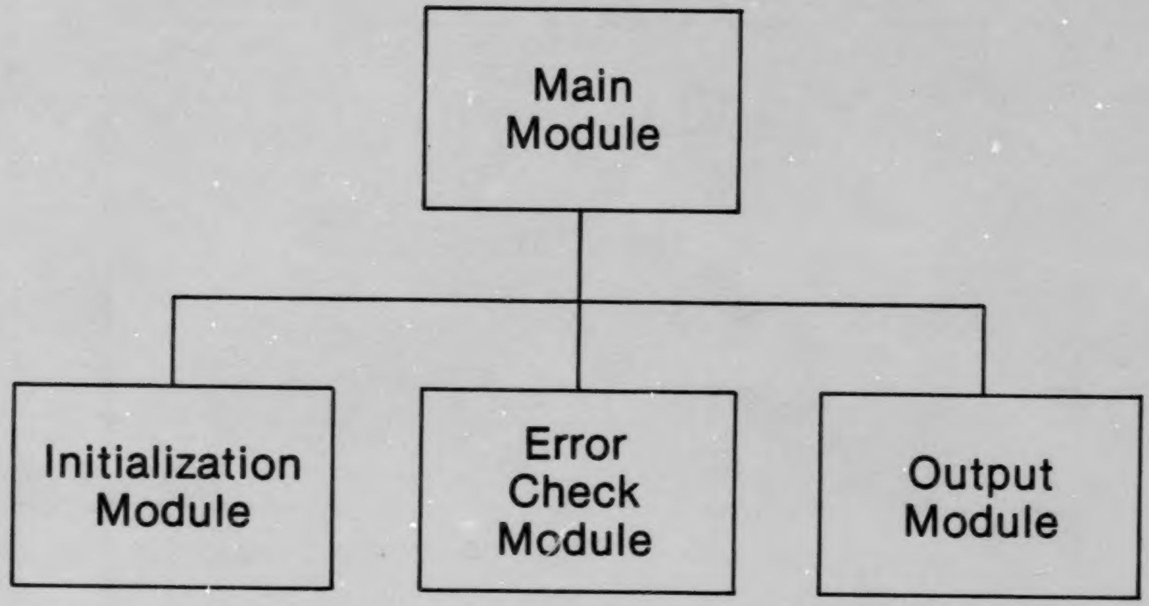
Address Line One Is? ____

Address Line Two Is? ____

Enter Current Balance or Zero ____

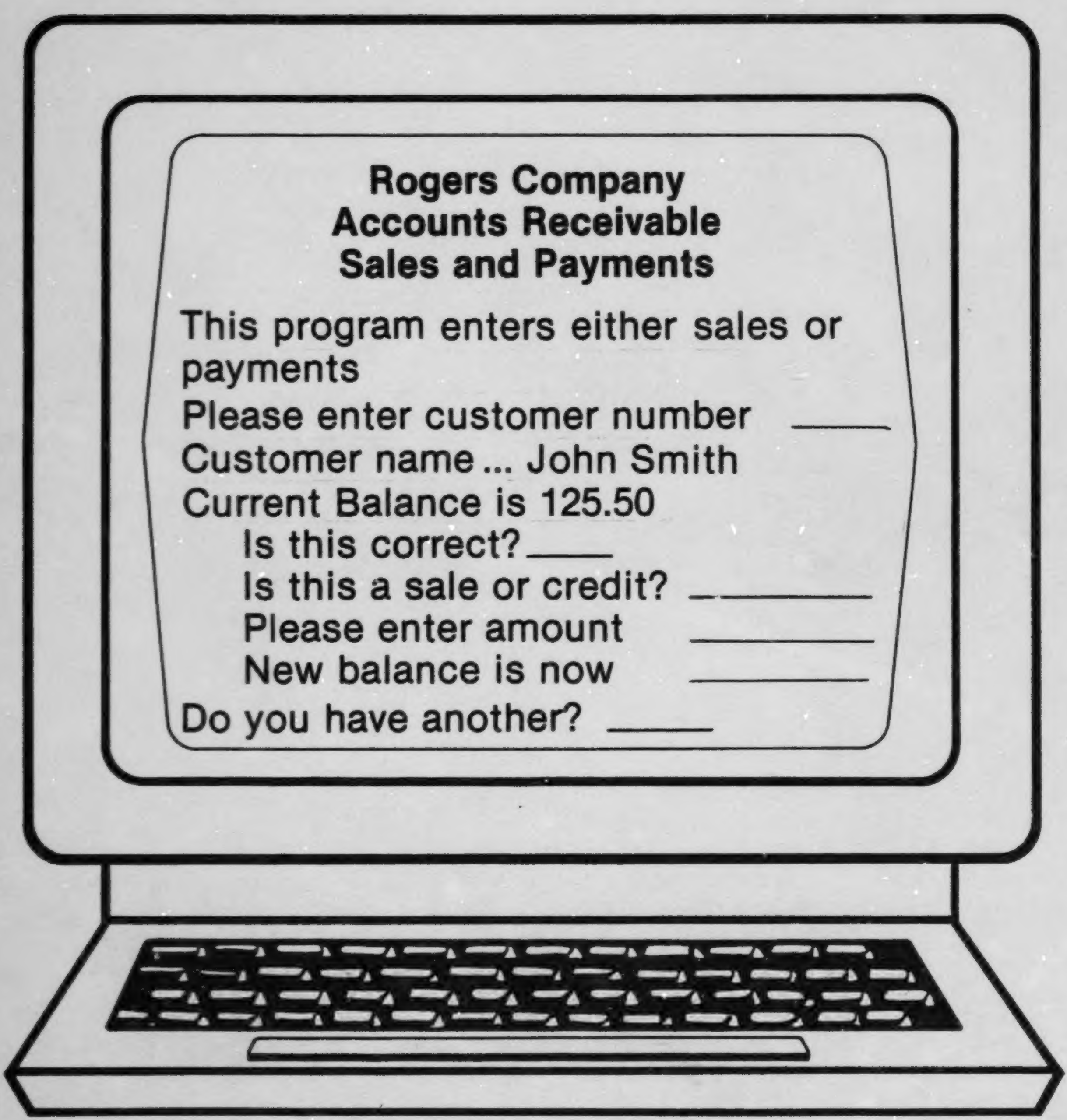
603

Hierarchy Chart for the New Customer Program, "RNEWCUS.BAS"


Main Module
Initialization Module
Error Check Module
Output Module

604

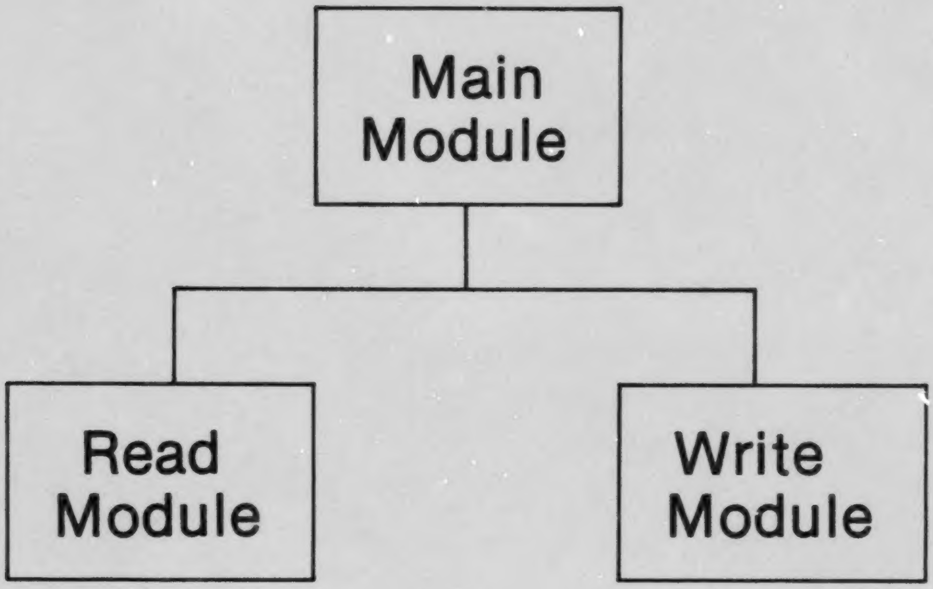
Design of the Sales and Payments Screen



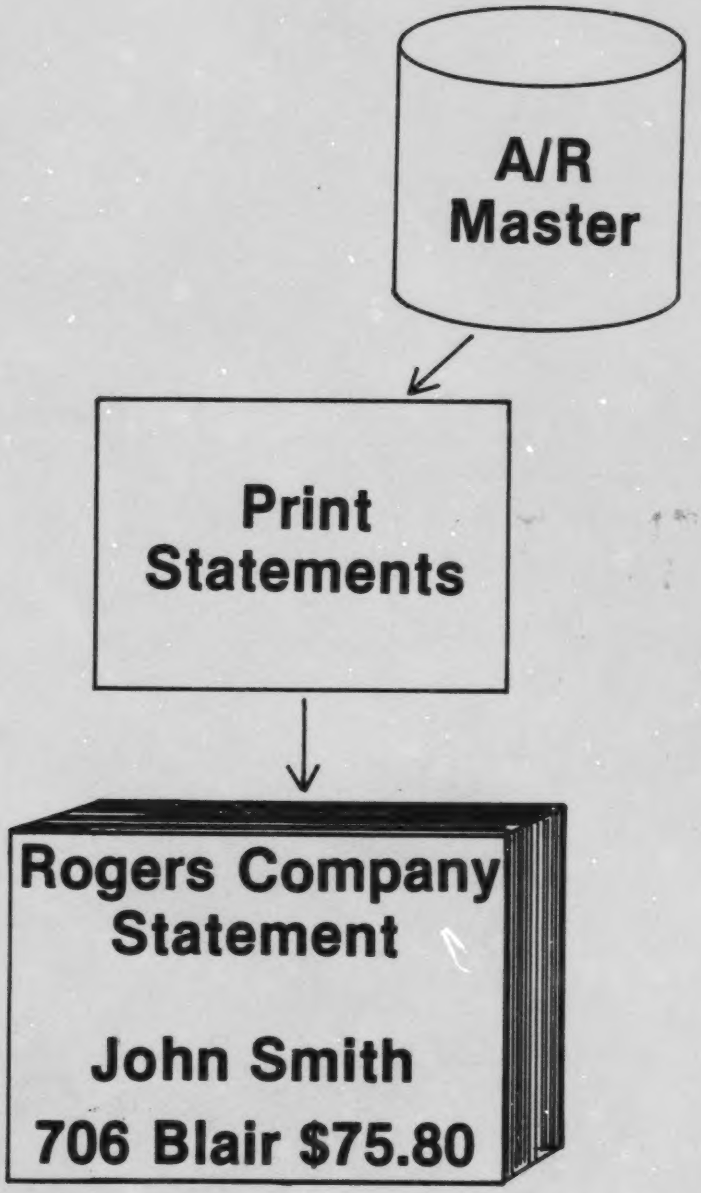
```
            Rogers Company
          Accounts Receivable
          Sales and Payments
This program enters either sales or
payments
Please enter customer number    _____
Customer name ... John Smith
Current Balance is 125.50
    Is this correct?_____
    Is this a sale or credit? ________
    Please enter amount       ________
    New balance is now        ________
Do you have another? _____
```

TM 8

Hierarchy Chart for the Sales and Payments Program


Main Module
Read Module
Write Module

Overview of the Print Statements Program



A/R Master

Print Statements

Rogers Company Statement

John Smith

706 Blair $75.80

BUSINESS DATA PROCESSING IN BASIC

UNIT XIV

ASSIGNMENT SHEET 1: BASIC File Commands

1. Write a BASIC command to open a file named RDAT.BAS, for relative access and buffer number three is to be used.

2. Write a BASIC CLOSE statement to close the file in question one above.

608

ASSIGNMENT SHEET 2: BASIC Buffer Commands

1. Write a correct FIELD statement for buffer three for the following data record:

Data	Length
Name	25
Address Line 1	30
Address Line 2	30
Address Line 3	30
Amt owed	6

BUSINESS DATA PROCESSING IN BASIC

UNIT XIV

ANSWERS TO ASSIGNMENT SHEETS

Assignment Sheet 1

1. Open "R", 3, "RDAT.BAS"
2. Close 3

Assignment Sheet 2

1. FIELD 3, 25 AS A1$, 30 AS A2$, 30 AS A3$, 30 A4$, 6 AS A5$

UNIT XIV

JOB SHEET 1: Print Program for Rogers Company Accounts Receivable

1. Finish the accounts receivable system for the Rogers Company by writing the print program to bill customers. Use the pseudocode in HO 6 and ask your instructor whether to design your program to print on the screen rather than the printer (to save time in your school's programming lab).

611

BUSINESS DATA PROCESSING IN BASIC

UNIT XIV

ANSWERS TO JOB SHEET

Job Sheet I

```
10          REM
20          REM
30          REM PROGRAM NAME IS RPRINT.BAS
40          OPEN "R", 1, "RAR.DAT"
50          FOR I = 1 TO 19
60          FIELD 1, 2 AS A1$, 30 AS A2$, 30 AS A3$, 30 AS A4$, 20 AS A5$, 7 AS A6$
70          GET 1,I
80          ID = CVI (A1$)
90                AMT = VAL (A6$)
100         GOSUB 160
110         NEXT I
120         INPUT"RETURN TO MAIN MENU"; YN$
130         IF YN$ = "N" GOTO 50
140         CLOSE
150         RUN "RMENU.BAS"
160         REM      *********  PRINT ROUTINE ********
170         IF CVI (A1$) = 0 GOTO 190
180         PRINT ID,A2$,A3$,A4$,A5$,AMT
190         RETURN
200         END
```

BUSINESS DATA PROCESSING IN BASIC

UNIT XIV

EXAMINATION

1. Match the phrases in Column A with the corresponding terms in Column B.

		Column A		Column B
____	a.	I/O area	1.	Balance forward
____	b.	Fetches a record	2.	FIELD
____	c.	Defines a buffer	3.	GET
____	d.	Simple form of accounts receivable	4.	Buffer
____	e.	Left justifies data	5.	LSET
____	f.	Calls on the operating system to run a program	6.	RUN

2. The following statements are either true or false. Circle "T" if the statement is true; circie "F" if the statement is false.

T F a. BASIC is the most widely used computer language.

T F b. Inventory applications are common in small business data processing.

T F c. The Rogers Company sample application uses a balance forward type of accounts receivable system.

T F d. The sales slip will be used to enter information for a particular customer account.

3. Explain how menus and prompts are used to help make the application user-friendly.

4. Write a correct BASIC statement to open a relative file named "INVEN.BAS" which will use buffer number two.

5. What BASIC language statement(s) in the main menu program in the Rogers Company sample paints the menu?

6. Explain the functions of any two modules in the new customer program.

7. List the steps which the payment/sales program goes through for the user.

BUSINESS DATA PROCESSING IN BASIC

UNIT XIV

ANSWERS TO EXAMINATION

1. a. 4
   b. 3
   c. 2
   d. 1
   e. 5
   f. 6

2. a. T
   b. T
   c. T
   d. T

3. Menus give a simple set of choices, which the user can obtain by simply entering the number of the choice. Prompts coach the user as to what information is to be inputted or what decisions are to be made.

4. OPEN "R", 2, "INVEN.BAS"

5. PRINT statements 30, 40, 50, 60, 70

6. a. Main module - prints main menu screen, controls input, controls other modules

   b. Initialization module - sets all the ID's on the file to zero to initialize them.

7. a. Accept customer I.D.
   b. Fetch customer's name and record.
   c. Print customer's name, balance, and ask for correction.
   d. Accept new amount; compute new balance.
   e. Write new balance on customer's record.

RPG II BUSINESS APPLICATION PROGRAMS

UNIT XV

UNIT OBJECTIVE

After completing this unit, the student should have reviewed subject matter presented during the first-year Data Processing course plus a review of exception output records.

SPECIFIC OBJECTIVES

After completion of this unit, students should be able to:

1.1. Define Accounts Receivable and list the purposes of controlling this application.

1.2. List basic functions of an automated receivable transaction register.

1.3. Write an RPG program to produce an Accounts Receivable Register.

2.1. Define file maintenance and list its functions.

2.2. Write an RPG program which provides updating for a Master Subscriber File.

3.1. List the objectives of a computerized billing system.

3.2. Write an RPG program to compute electric bills.

4.1. Identify the benefits of automated payroll and list examples of deductions commonly taken.

4.2. Write an RPG program to compute payroll deductions for employees.

RPG II BUSINESS APPLICATION PROGRAMS

UNIT XV

SUGGESTED ACTIVITIES

I. Instructor

   A. Provide students with objectives and discuss.

   B. Provide students with information and assignment sheets and discuss.

   C. Discuss unit and specific objectives.

   D. Discuss information sheets.

   E. Provide handouts, assignment sheets, and job sheets.

   F. Discuss assignment sheets.

II. Student

   A. Read objectives.

   B. Study information sheets and handouts.

   C. Complete assignment and job sheets.

INSTRUCTIONAL MATERIALS

I. Unit Content

   A. Objectives

   B. Notes to the instructor

   C. Information sheets

   D. Handouts

      1. HO 1: Overview of Assignment Sheets 1, 2, 3, and 4

      2. HO 2: Overview of Accounts Receivable Register Problem

      3. HO 3: Overview of Master Subscriber File Problem

      4. HO 4: Overview of Electric Bill Calculations Problem

      5. HO 5: Overview of Employee Payroll Deductions

E. Assignment sheets

  1. AS 1: Accounts Receivable Register
  2. AS 2: Master Subscriber File Update
  3. AS 3: Electric Bill Calculations
  4. AS 4: Employee Payroll Deductions

F. Answers to assignment sheets

G. Job sheets

  1. JS 1: Execute Accounts Receivable Register on Computer
  2. JS 2: Execute Master Subscriber File Program
  3. JS 3: Execute Electric Bill Problem
  4. JS 4: Execute Payroll Calculation Program

H. Examination

I. Answers to examination

II. References

A. IBM System 34/36/38 RPG II Reference Manual. Information Development. Rochester, MN: IBM Corp., 1983.

B. Shelly, Gary B. and Thomas J. Cashman. Computer Programming RPG II. Fullerton, CA: Anaheim Publishing Company, 1978.

C. IBM RPG II Self Study. (Inquire at Guided Learning Center of your local IBM branch office for availability of classes and textbooks.)

RPG II BUSINESS APPLICATION PROGRAMS

UNIT XV

NOTES TO THE INSTRUCTOR

I. Directions, suggestions, and necessary materials

A. Directions: Have your students code a solution for each of the data processing problems in this unit. The students will need blank coding sheets of all types in order to complete this set of exercises. They will also need a blank Print Chart for problem number 4.

B. Suggestions: It is suggested that the students concentrate on one problem at a time. Have students examine all output shown; then read and reread the problem statement until they are sure about what needs to be coded. Then proceed in this manner:

1. Describe the input and output files on a File Description sheet.
2. Describe each type of input record and field on an Input sheet.
3. Describe the desired output on an Output sheet.
4. Describe calculation entries on a Calculation sheet. Remember to place all detail-time calculation entries ahead of all total-time entries on this sheet.
5. Reread the problem, examine all illustrations, and review all of your coding.

C. Materials: In addition to this text, you will need a number of blank RPG II specification sheets and the RPG II language manual. Also, get one or two blank Print Chart sheets.

II. Overview of Assignment Sheets 1, 2, 3, and 4

As the name implies, this is a unit in which practice is provided for the students so that they can review the principles of problem solving and apply the rules for coding solutions, using whatever entries are appropriate to get each job done. Four assignment sheets are presented in this unit.

1. In this unit, AS 1 asks the student to print an "Accounts Receivable Register" from a file of accounts receivable records found on disk.
2. AS 2 asks the student to update the records in a "Master Subscriber" file by entering records from the console keyboard that either has renewals or changes of address. The master subscriber file is an indexed file which includes a list of the keyed transactions.

3. AS 3 asks the student to compute monthly electric bills for customers.

4. AS 4 asks the student to create a new sequential file of payroll records by copying the payroll records in an existing indexed file. While this is being done, the student verifies each payroll record. If any record includes too many deductions (which would result in either no net pay or worse, negative net pay), information is printed about that record as a part of an exception report.

RPG II BUSINESS APPLICATION PROGRAMS

UNIT XV

INFORMATION SHEET

I. Automated Accounts Receivable (HO 1 and HO 2, AS 1; JS 1)

  A. Definition - automated accounts receivable is the amount of money owed to a company by its customers for merchandise sold or services rendered on a credit basis.

  B. Purposes of controlling A/R

    1. To facilitate collection of money owed - cash flow

    2. To minimize losses from bad debts

    3. To maintain customer goodwill through prompt and accurate recordkeeping

  C. Functions of an Automated Accounts Receivable (A/R) Transaction Register

    1. Runs daily

    2. Copies transactions affecting accounts receivable

    3. Uses invoice summary file

    4. Confirms receipts and invoices (Report included in this unit shows only invoice amounts.)

    5. Reports sales amount vs. income amount

    Note: A/R Register would be more detailed in true environment showing sales amount vs. income amount.

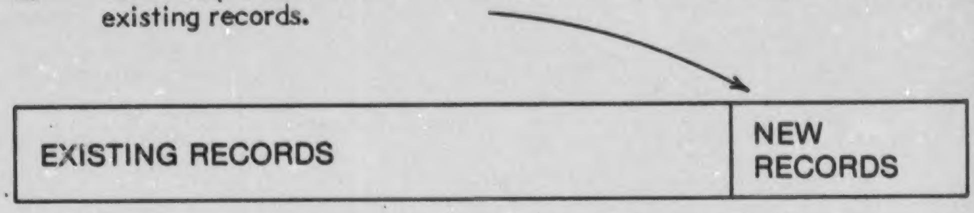
II. File Maintenance for a Master Subscriber File (HO 3; AS 2; JS 2)

  A. Definition - file maintenance is performing functions to keep a file current for daily processing needs, such as adding a new customer or updating an account balance.

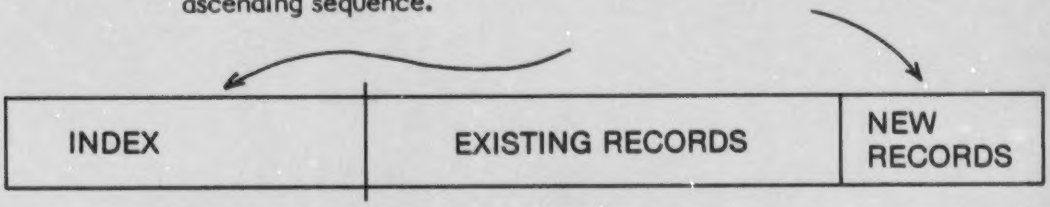
  B. Functions

    1. Adding records to a file - the file maintenance function is used to add new records to those in an existing file.

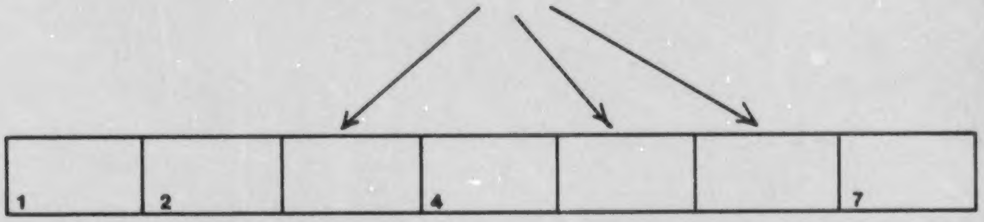
a. To a Sequential File: new records are placed behind the existing records.



b. To an Indexed File: new records are placed behind the existing records and new index entries are merged into ascending sequence.



c. To a Direct File: new records are assigned relative record numbers of unused (reserved) spaces.



2. Updating records - the file maintenance function in which the contents of one or more fields of data in a record are to be changed.

   a. Example: "I have just moved. Please change my address to 12 Third St.; Newtown, OK 37115. My customer number is 4041."

   b. Before updating record

A	4041	ME	BOX 75	WESTOP, AK 67811

   c. After updating record

A	4041	ME	[illegible] THIRD ST.	NEWTOWN, OK 37115

3. Tagging/flagging for deletion - the file maintenance function used to identify those records in a file that shall no longer be active.

    a. Flagging records for deletion - same as "tagging" records for deletion.

    b. In every case, the record to be tagged for deletion is "updated" by changing the contents of a code field. For example, active records might contain the letter "A" in this code field while records tagged for deletion might contain the letter "D" in the same field after the updating takes place.

III. Objectives of computerized billing system (HO 4; AS 3; JS 3)

A. Record and acknowledge customer electric bills

B. Compute prices, discounts, and totals (taxes, if applicable)

C. Automatically produce timely and accurate customer bills through cyclic billing during the month

Note: Cyclic billing refers to the weekly billing of customers segmented alphabetically into four different groups.

D. Provide necessary management reports and inquiry capabilities for their customers, i.e., past due customer reports and individual customer states

E. Reduce paper flow throughout office

F. Reduce clerical time and errors from beginning to end of billing cycle

IV. Automated payroll benefits (HO 5; AS 4; JS 4)

A. Analyze and control company payroll costs

B. Allow for clerical savings

C. Facilitate audit procedures

D. Streamline typical payroll deductions - amount subtracted from an employee's earnings before payment is made to the employee by the employer

1. Federal tax
2. FICA tax
3. State tax
4. Local tax, if any
5. Union deductions
6. Savings bonds
7. Life insurance
8. Hospitalization

RPG II BUSINESS APPLICATION PROGRAMS

UNIT XV

HANDOUT I: Overview of Assignment Sheets 1, 2, 3, and 4

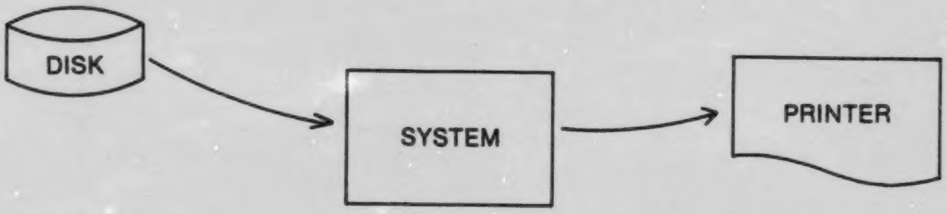
When doing the assignment sheets in this unit, concentrate on one problem at a time. Examine all output shown; then read and reread the problem statement until you are sure about what needs to be coded. Then proceed in this manner:

1. Describe the input and output files on a File Description sheet.
2. Describe each type of input record and field on an Input sheet.
3. Describe the desired output on an Output sheet.
4. Describe calculation entries on a Calculation sheet. Remember to place all detail-time calculation entries ahead of all total-time entries on this sheet.
5. Reread the problem, examine all illustrations, and review all of your coding.

In addition to this guide, you will need a number of blank RPG II specification sheets and the RPG II language manual. Also, get one or two blank Print Chart sheets.

HANDOUT 2: Overview of Accounts Receivable Register Problem

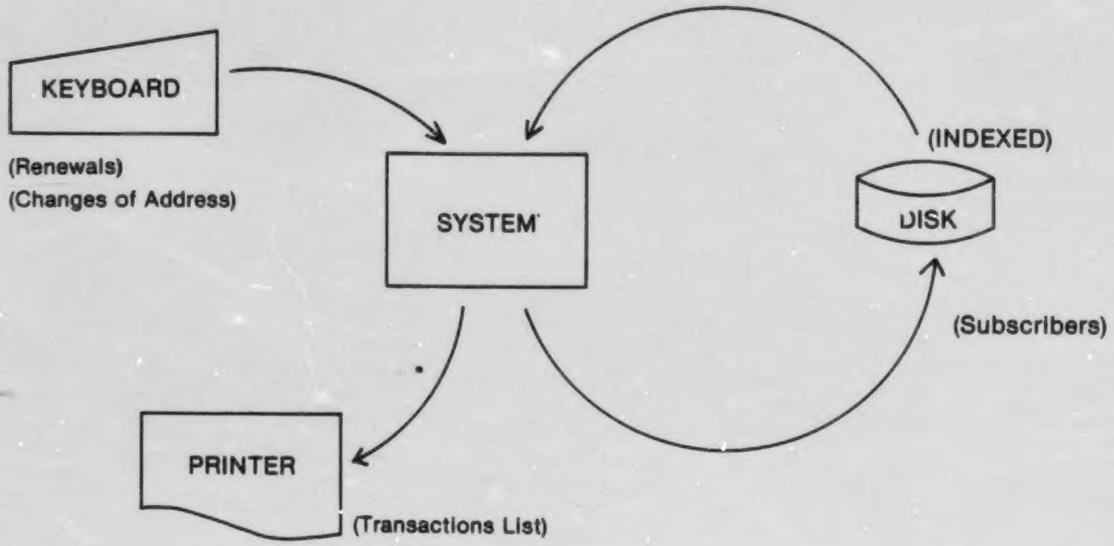
As the name implies, this is a unit in which practice is provided for the students so that they can review the principles of problem solving and apply the rules for coding solutions, using whatever entries are appropriate to complete each job done. The first exercise is concerned with Accounts Receivable. The diagram below shows the steps of the process.



After examining the diagram above, turn to AS I to write an RPG program to produce an Accounts Receivable Register.

HANDOUT 3: Overview of Master Subscriber File Problem

The Master Subscriber File is an indexed file. The diagram below shows the process flow for this procedure.


KEYBOARD
(Renewals)
(Changes of Address)
SYSTEM
(INDEXED)
DISK
(Subscribers)
PRINTER
(Transactions List)

Here are some new rules for reference when describing an indexed file.

1. Length of key field (29-30) - specify the length of the field that is to be used as the key field.
   a. Omit the leading zero.
   b. The maximum length of a key field is 29 characters if the data is alphameric.
   c. The maximum length is 15 characters if it is a numeric field.
2. Record Address Type (31) - Enter code "A." This code is used when processing or creating (loading) indexed files.
3. Type of File Organization (32) - Enter code "I" to designate Indexed file organization.

4. Key Field Starting Location (35-38) - Specify the starting location of the key field in the disk record.

   a. Starting location is the leftmost position.

   b. Omit leading zeros.

If additional information is needed on the Chain Operation, refer to RPG II Reference Manual.

Turn to AS 2 for directions and information on updating the records in a Master Subscriber file by entering records from the console keyboard that will either renew or change addresses.

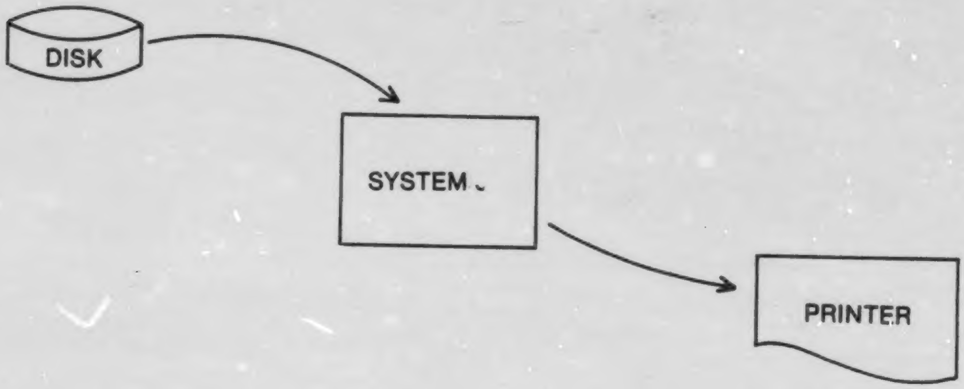
627

RPG II BUSINESS APPLICATION PROGRAMS

UNIT XV

HANDOUT 4: Overview of Electric Bill Calculations

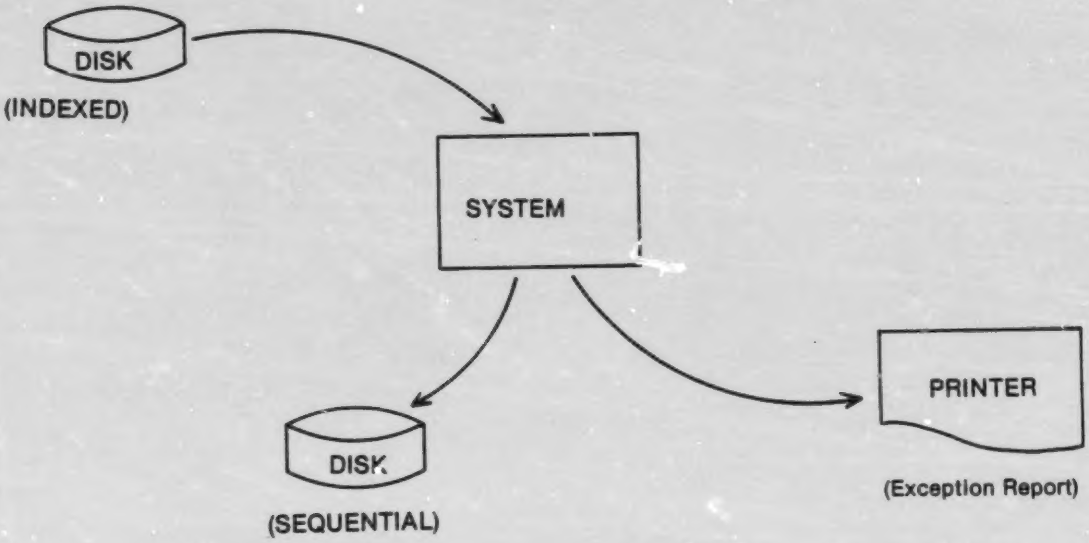
In this exercise you will be computing monthly electric bills for cutomers. The diagram below shows the process flow for this procedure.



Turn to AS 3 for directions and information on calculating a customer's electric usage bill.

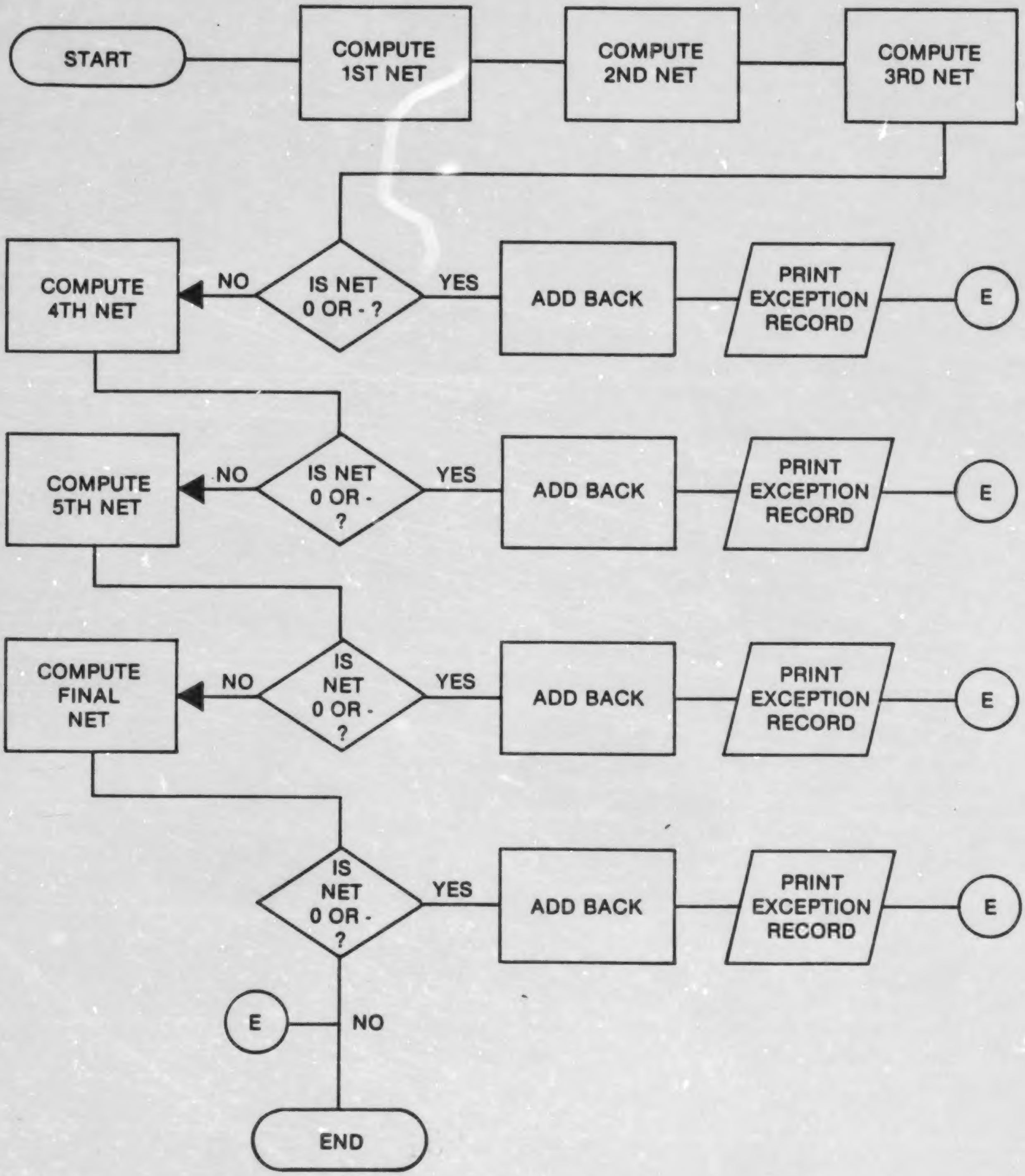
HANDOUT 5: Overview of Employee Payroll Deductions

In this exercise you will create a new sequential file of payroll records by copying the payroll records in an existing indexed file. While this is being done, verify each payroll record. If any record includes too many deductions (which would result in either no net pay or worse, negative net pay) print information about that record as part of an exception report. The diagram below shows the steps of this process.

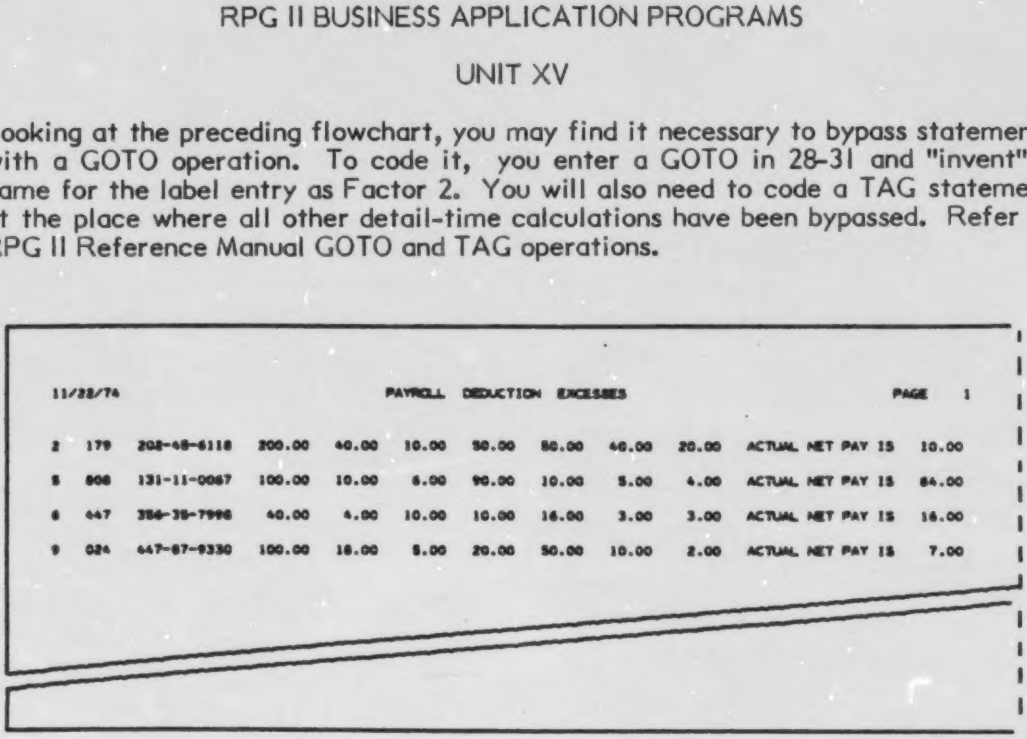


The flowchart below is also for your reference. In order to compute the net pay value for a record, remember this series of steps:

1. 1st Net Pay = Gross Pay - Federal Income Tax
2. 2nd Net Pay = First Net - FICA (social security payment)
3. 3rd Net Pay = 2nd Net - Deduction 1
4. 4th Net Pay = 3rd Net - Deduction 2
5. 5th Net Pay = 4th Net - Deduction 3
6. Final Net Pay = 5th Net - Deduction 4


START
COMPUTE 1ST NET
COMPUTE 2ND NET
COMPUTE 3RD NET
IS NET 0 OR - ?
NO
YES
COMPUTE 4TH NET
ADD BACK
PRINT EXCEPTION RECORD
E
IS NET 0 OR - ?
NO
YES
COMPUTE 5TH NET
ADD BACK
PRINT EXCEPTION RECORD
E
IS NET 0 OR - ?
NO
YES
COMPUTE FINAL NET
ADD BACK
PRINT EXCEPTION RECORD
E
IS NET 0 OR - ?
YES
ADD BACK
PRINT EXCEPTION RECORD
E
E
NO
END

Looking at the preceding flowchart, you may find it necessary to bypass statements with a GOTO operation. To code it, you enter a GOTO in 28-31 and "invent" a name for the label entry as Factor 2. You will also need to code a TAG statement at the place where all other detail-time calculations have been bypassed. Refer to RPG II Reference Manual GOTO and TAG operations.



```
11/22/74                                    PAYROLL  DEDUCTION  EXCESSES                        PAGE    1

2  179  202-48-6118  200.00  40.00  10.00  50.00  80.00  40.00  20.00  ACTUAL NET PAY IS  10.00
5  808  131-11-0067  100.00  10.00   6.00  90.00  10.00   5.00   4.00  ACTUAL NET PAY IS  84.00
6  447  356-35-7996   40.00   4.00  10.00  10.00  16.00   3.00   3.00  ACTUAL NET PAY IS  16.00
9  024  447-87-9330  100.00  18.00   5.00  20.00  50.00  10.00   2.00  ACTUAL NET PAY IS   7.00
```

Get a blank Print Chart and fill it out for use as a worksheet. Here are suggestions for the use of this chart. First, identify from the report sample the number of different heading lines to be printed. In this example we have only one, the report heading line. Next, print the letter H to the left of the number 6 that identifies line 6 on the print chart. Then identify the number of different body lines to be printed. In this example, all body lines are alike in format, so you need to plan for only 1 body line. In our example, this body line represents an exception record format and you will print the letter E to the left of the number 9 on line 9.

Now you are ready to fill in the blanks. Since the exception records must include many fields and some messages, plan it first.

Also remember that the system date field named UPDATE and the page numbering field named PAGE are special field names reserved for RPG II use.

Turn to AS 4 to write an RPG program to compute payroll deductions for employees.

UNIT XV

ASSIGNMENT SHEET I: Accounts Receivable Register Problem

Directions: Read the following information.

A file of 80-character disk records contains information about a certain company's accounts receivable. An "Accounts Receivable Register" is printed for management each month so that it can be determined which invoices have not yet been paid by customers who purchased goods from the company. A sample of that report is on this page.

```
                              ACCOUNTS  RECEIVABLE  REGISTER

CUSTOMER CUSTOMER NAME            STATE   CITY   INVOICE    INVOICE       INVOICE
 NUMBER                                           NUMBER     DATE          AMOUNT

  1281    AMERICAN STEEL CO        36      49     11666     11/23/67       640.31

  1281    AMERICAN STEEL CO        36      49     12336     12/30/67       909.04

  2179    APALACHIN LUMBER CO       4     227      9852      9/15/67       469.20

  2283    B J E SERVICE CORP       22      37     12332     12/29/67     1,474.78

 11905    CHALLIS ALMERS           47      77     10901     10/18/67        27.63

 29031  . DENNIS MFG CO             6      63     11615     11/14/67       440.12


                                                                      17,524.23 *
```

The input records contain these fields in the positions shown.

Positions	Field
8-29	Customer Name
34-38	Invoice Number
39-43	Customer Number
44-45	State (number code)
46-48	City (number code)
49-54	Invoice Date
74-80	Invoice Amount

Using the information above and the sample report, write an RPG program to produce an Accounts Receivable Register.

After you complete the writing of the RPG program, turn to JS I to execute your program on the computer.

RPG II BUSINESS APPLICATION PROGRAMS

UNIT XV

ASSIGNMENT SHEET 2: Master Subscriber File Update

Directions: Read the following information and code up the solution. A master file of subscriber records is stored as an indexed disk file. Each record is 85 characters long and includes a key field in positions 1-7. The fields of information in the master file records are:

Positions	Field
1-7	Subscriber Number (key field)
8-31	Subscriber Name
32-55	Street Address
56-79	City, State, and Zip
80-85	Expiration Date of Subscription

The records in this file are updated on a daily basis for 2 kinds of transactions:

1. renewal of subscription
2. change of address

The changes are keyed in from the console as an Interactive Data Entry file and contain these fields:

Type	Positions	Field
Renewal	1	Code - letter R
	2-8	Subscriber Number
	9-10	Number of Years Renewed
Change of Address	1	Code - letter C
	2-8	Subscriber Number
	9-32	New Street Address
	33-56	New City, State, and Zip

You are to code a solution to this problem so that these requirements are met:

1. The master file is to be updated randomly using "chaining."
2. For a "renewal," add the number of years in the renewal record to the "year" position of the expiration date field in the master record.
3. For a "change of address," replace the street address and the city, state, and zip code fields in the master record.
4. Include a simple Auto Report list of the keyed in transactions. Here's a sample of that listing:

```
CODE     SUBSCRIBER  YEARS     STREET                     CITY/STATE

---      -------     --

---      -------               --------------             ----------

---      -------     --
```

5. If no master record is found to match either a renewal or change of address record, print the message, NO MASTER FOUND.

6. For printing, specify the edit code Z (zero suppress only) for subscriber number and for number of years to be renewed.

7. Include a "program identification" entry at the top of each page. You may use any name provided it starts with a letter. It may be up to six characters long.

When you are ready to code a solution, you may refer to the RPG II reference manual if additional information is needed.

ASSIGNMENT SHEET 3: Electric Bill Calculations

Directions: Read the following information. The calculations you are to describe shall be those used to compute a customer's electric usage bill. This particular electric company has the following rates:

Usage	Rate
1-50 kwh	$.05 each
51-100 kwh	$.04 each over the 50, + $2.50
101-300 kwh	$.03 each over the first 100, + $4.50
over 300 kwh	$.025 each over the first 300, + $10.50

Here are some examples:

Customer	Usage	Monthly Bill
123	30 kwh	?
776	70 kwh	?
406	120 kwh	?
882	310 kwh	?
396	50 kwh	?
451	100 kwh	?
275	300 kwh	?
666	0 kwh	?

The monthly bill for customer #123 comes to $1.50 for the 30 kwh of electricity used. From the table, the rate is $.05 each.

The monthly bill for customer #776 comes to $3.30 for the 70 kwh used. From the table, the rate is $.04 each over 50 + $2.50.

1. Use a piece of scratch paper to determine the bill for each customer. Round all answers to the nearest whole cent.

   NOTE: Check you answers against these. If you disagree, recalculate the bill or bills in question before you continue.

2. Your billing calculations depend upon the meter readings taken last month and this month because the number of kwh used equals the reading this month less the reading last month. Normally this is no particular problem to compute; but every so often, the meter "turns over," that is, the number shown passes the all 9's reading. When this happens, you need to imagine that a 1 is carried into the next place when you compute usage. Keep in mind that computers don't "imagine" things like that, they need to be programmed to include the necessary steps to actually perform the calculation.

Here are two examples: Place answers on scratch paper.

This Reading	Last Reading	Usage
02830	02750	____
00150	99900	____

3. You will refer to the following list of steps that are needed to code when you begin to describe the calculations:

    1. Compute the usage for the month.

        a. Subtract last month's reading from this month's reading.

        b. If the answer is a negative value, we know that the meter went past all 9's at some time during the month.

        c. To correct this answer, increase this month's reading by 100000 and then subtract last month's reading from this larger value.

    2. If the usage is up to and including 50 kwh, multiply the answer from step 1 by $.05 to compute the bill.

    3. If the usage is from 51 to 100 kwh inclusive, find out how many kwh over 50, multiply them by $.04, and then add $2.50 to compute the bill.

    4. If the usage is from 101 to 300 kwh inclusive, find out how many kwh over 100, multiply them by $.03, and then add $4.50 to compute the bill.

    5. If the usage is over 300 kwh, find out by how many, multiply them by $.025 (or 2.5¢) and round the answer to the nearest cent. Then add $10.50 to compute the bill.

Before you can code the solution to this problem, you need to know:

1. The field that will print the amount of the bill is named BILL in the output specifications. It is to be 5 positions long with 2 decimal positions.

2. The field that will print the usage during the month is called KWH. It is to be 5 positions long with no decimals.

3. The numeric input fields needed are defined as follows:

Field	Size	Purpose
NEW	5, no decimals	holds this month's meter reading
LAST	5, no decimals	holds last month's meter reading

Now, take a blank Calculation sheet and specify the statement needed to compute the usage for the month. Call the result USE with 5 positions, no decimals. Also perform any other necessary calculations to do steps 2 through 5.

4. Now code up appropriate File Description, Input, and Output specification sheets.

When you have completed the coding process, turn to JS 3 to execute your program on the computer.

RPG II BUSINESS APPLICATION PROGRAMS

UNIT XV

ASSIGNMENT SHEET 4: Employee Payroll Deductions

Directions: Read the following information and code the solution. An indexed disk file of 100-character records is to be copied for reference. Its key field is in positions 1-8. The new file will contain identical records but is to be organized sequentially.

As each input record is read, a computation for net pay is to be performed by following this formula:

Net Pay = Gross Pay - Federal Income Tax - Social Security Payment - Deduction 1 - Deduction 2 - Deduction 3 - Deduction 4...

The last four items (deduction payments 1-4) have been authorized by each employee and are to be taken out each pay period, unless by taking it out the net pay value becomes either zero or negative. If that should happen, you are directed to add back the deduction amount just subtracted, print an exception record right at that point in the calculations, and bypass all further net pay calculations for that record.

You will need blank coding forms of each type. You may need 2 or 3 Calculation sheets and 2 or 3 Output sheets in addition to the File Description and Input sheets.

Code this problem solution in a couple of phases rather than all at once. First, let's concentrate on making a copy of the indexed file records. The output file is to contain the same information but shall be sequentially organized rather than indexed. If you think you have enough information, code all of the entries needed to copy this file.

Wait a minute. The fields of data needed to compute net pay are not specified as yet. Use your input sheet and make these additional entries. You are to make up field names for them.

Positions	Decimals	Field	
1-3	0	Department Number	Key Field
4-8	0	Man Number	
9-17	0	Social Security Number	
18-22	2	Gross Pay	
23-27	2	Federal Income Tax	
28-32	2	Social Security Tax Amount	
53-56	2	First Deduction Amount	
57-60	2	Second Deduction Amount	
61-64	2	Third Deduction Amount	
65-68	2	Fourth Deduction Amount	

We are now ready to describe the output records for the report that is to show for which employees no deductions, or only some, could be taken. Each of these records is an exception to the normal payroll calculation, and it has been determined by management that the list of exception records shall include:

1. Some information to identify the employee
    a. department number
    b. employee number
    c. social security number
2. The amount of money earned this pay period
    a. gross pay
3. The tax amounts deducted
    a. federal income tax
    b. social security tax
4. The authorized deductions amounts
    a. deduction 1
    b. deduction 2
    c. deduction 3
    d. deduction 4

NOTE: If deduction 4 can also be taken, no exception record is printed.

5. A message that tells which, if any, deductions were taken
    a. NO DEDUCTIONS TAKEN
    b. INCLUDING D1
    c. INCLUDING D1, D2
    d. INCLUDING D1, D2, D3

NOTE: Refer to HO 5.

To assure yourself that your calculations satisfy the job requirements, use the following test data and "run it" through your program sequence. Use a piece of scratch paper to keep track of:

1. The setting of each assigned indicator
2. The value of the net pay field at each step.

#	Gross	FIT	FICA	D1	D2	D3	D4	Net Pay	Deductions
1	100	20	5	20	1	2	3	$ 49	All
2	200	40	10	50	50	40	20	$ 10	D1, D2, D3
3	100	18	5	20	50	10	2	$ 7	D1, D2
4	200	45	10	70	80	10	10	$ 75	D1
5	100	10	6	90	10	5	4	$ 84	None

Note: the abbreviation FICA is used for Social Security payments

BEST COPY AVAILABLE

RPG II BUSINESS APPLICATON PROGRAMS
UNIT XV

ANSWERS TO ASSIGNMENT SHEETS

Assignment Sheet I

File Description Specifications

Line	Form Type	Filename	File Type (I/O/U/C/D)	File Designation (P/S/C/R/T/D)	Record Length	Device
02	F	ACCTSREC	I	P	80	DISK
03	F	REGISTR	O		132	PRINTER
04	F					

RPG INPUT SPECIFICATIONS

Line	Form Type	Filename or Record Name	Sequence	Record Identifying Indicator	Field Location From	Field Location To	Decimal Positions	RPG Field Name
01	I	ACCTSREC	NS	01				
02	I				8	29		CNAME
03	I				34	38		INNO
04	I				39	43		CUSNO
05	I				44	45		STATE
06	I				46	48		CITY
07	I				49	54	0	INDATE
08	I				74	80	2	IWAMT
09	I							
10	I							
11	I							

IBM International Business Machines Corporation

RPG CALCULATION SPECIFICATIONS

Program

Programmer

Date

Keying Instruction

Graphic

Key

Card Electro Number

Line	Form Type	Control Level	Indicators	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Half Adjust (H)
01	C		01	IWAMT	ADD	TOTIWY	TOTIWV	8	2	
02	C									

Assignment Sheet I (con't)

BEST COPY AVAILABLE

IBM International Business Machines Corporation

RPG OUTPUT SPECIFICATIONS

Program
Programmer
Date
Keying Instruction: Graphic / Key
Card Electro Number

Line	Form Type	Filename or Record Name	And/Or	Type (H/D/T/E)	Space After	Output Indicators	Field Name or EXCPT Name	Edit Codes	End Position in Output Record	Constant or Edit Word
01	O	REGISTR		H	2	1P				
02	O		OR			OF				
03	O								48	'ACCOUNTS RECEIVABLE'
04	O								58	'REGISTER'
05	O			H	1	1P				
06	O		OR			OF				
07	O								28	'CUSTOMER CUSTOMER NAME'
08	O								57	'STATE CITY INVOICE'
09	O								80	'INVOICE INVOICE'
10	O			H	2	1P				
11	O		OR			OF				
12	O								13	'NUMBER'
13	O								66	'NUMBER DATE'
14	O								80	'AMOUNT'
15	O			D	1	01				
16	O						CUSNO		12	
17	O						CNAME		35	
18	O						STATE		40	
19	O						CITY		47	
20	O						INNO		56	
	O						INDATE	Y	68	
	O						INAMT	2	80	
	O			T	3	LR				
	O						TOTINV		80	
	O								82	'*'

641

BEST COPY AVAILABLE

RPG II BUSINESS APPLICATION PROGRAMS

UNIT XV

Assignment Sheet 2

File Description Specifications

Line	Form Type	Filename	File Type I/O/U/C/D	File Designation P/S/C/R/T/D/F	End of File E	Sequence A/D	File Format F/V/S/M/D/E	Block Length	Record Length	Mode of Processing L/R	Length of Key Field or of Record Address Field	Record Address Type A/P/I/K	Type of File Organization or Additional Area I/X/D/T/R/1 or 2	Overflow Indicator	Key Field Starting Location	Extension Code E/L	Device	Symbolic Device	Labels S/N/E/M	Name of Label Exit
02	F	CHANGES	I	P				62	60								CONSOLE			
03	F	SUBMAST	U	C					85	R	7	A	I		1		DISK			
04	F	LISTOUT	O						132								PRINTER			
05	F																			

IBM RPG INPUT SPECIFICATIONS

International Business Machines Corporation

Program
Programmer
Date
Keying Instruction: Graphic / Key
Card Electro Number

Line	Form Type	Filename or Record Name	Sequence	Number (1/N), E	Option (O), U, S	Record Identifying Indicator, **, or DS	1 Position	1 Not (N)	1 C/Z/D	1 Character	2 Position	2 Not (N)	2 C/Z/D	2 Character	3 Position	3 Not (N)	3 C/Z/D	3 Character	Stacker Select	P/B/L/R	Field Location From	Field Location To	Decimal Positions	RPG Field Name
01	I	CHANGES	NS			01	1		C	R														
02	I																				1	1		CODE
03	I																				2	8	0	SUBNO
04	I																				9	10	0	YEARS
05	I		NS			02	1		C	C														
06	I																				1	1		CODE
07	I																				2	8	0	SUBNO
08	I																				9	32		NEWST
09	I																				33	56		NEWCTY
10	I	SUBMAST	NS			03																		
11	I																				84	85	0	EXDATE

Assignment Sheet 2 (con't)

BEST COPY AVAILABLE

IBM International Business Machines Corporation

RPG CALCULATION SPECIFICATIONS

Program		Keying Instruction	Graphic
Programmer	Date		Key

Card Electro Number

Line	Form Type	Control Level (L0-L9, LR, SR, AN/OR)	Indicators Not	Indicators	And Not	And	And Not	And	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Half Adjust (H)	Resulting Indicators High (1>2) (Plus)	Low (1<2) (Minus)	Equal (1=2) (Zero)
01	C					01			SUBNO	CHAIN	SUBMAST					11		
02	C			01	N	11			EXDATE	ADD	YEARS	EXDATE						
03	C			02					SUBNO	CHAIN	SUBMAST					22		
04	C																	

IBM International Business Machines Corporation

RPG OUTPUT SPECIFICATIONS

Program		Keying Instruction	Graphic
Programmer	Date		Key

Card Electro Number

Line	Form Type	Filename or Record Name	AND/OR	Type (H/D/T/E)	Space Before	Space After	Skip Before	Skip After	Output Indicators Not	Indicator	And Not	And	And Not	And	Field Name or EXCPT Name	Edit Codes	Blank After (B)	End Position in Output Record	P/B/L/R	Constant or Edit Word
01	O	SUBMAST		D						01	N	11								
02	O														EXDATE			85		
03	O			D						02	N	22								
04	O														NEWST			55		
05	O														NEWCTY			79		
06	O	LISTOUT		H		2				1P										
07	O		OR							OF										
08	O																	20		'CODE SUBSCRIBER'
09	O																	37		'YEARS STREET'
10	O																	63		'CITY/STATE'
11	O			D		2				01										
12	O		OR							02										
13	O														CODE					
14	O														SUBSNO2					
15	O									01					YEARS					
16	O									01		11								'NO MASTER FOUND'
17	O									02					NEWST					
18	O									02					NEWCTY					
19	O									02		22								'NO MASTER FOUND'
20	O																			

Assignment Sheet 3

1.

Customer	Usage (kwh)	Calculation	Bill
123	30	30 x .05	$ 1.50
776	70	20 x .04 + 2.50	$ 3.30
406	120	20 x .03 + 4.50	$ 5.10
882	310	10 x .025 + 10.50	$ 10.75
396	50	50 x .05	$ 2.50
451	100	50 x .04 + 2.50	$ 4.50
275	300	200 x .03 + 4.50	$ 10.50
666	0	0 x .05	$ 0.00

2. a. 80
   b. 250

For the first example, we simply subtract the last reading (02750) from this reading (02830) to compute usage.

For the second example, you really need to tell the computer to subtract 99900 (last reading) from 100150 (this reading plus a 1 carry) in order to get the right usage.

Assignment Sheet 3 (con't)

3.

IBM International Business Machines Corporation

RPG CALCULATION SPECIFICATIONS

Program | Keying Instruction | Graphic | Card Electro Number
Programmer | Date | Key

Line	Form Type	Control Level (L0-L9, LR, SR, AN/OR)	Indicators	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Half Adjust (H)	Resulting Indicators Plus / High / 1 > 2	Minus / Low / 1 < 2	Zero / Equal / 1 = 2
01	C			NEW	SUB		USE	5	0			20	
02	C		20	NEW	ADD		BIG	6	0				
03	C		20	BIG	SUB		USE						
04	C		20		Z-ADD	0	BIG						
05	C				Z-ADD	USE	KWH	5	0				
06	C			USE	COMP	50					30		
07	C		N30	USE	MULT	.05	BILL	5	2				
08	C		N30		GOTO	END							
09	C			USE	COMP	100					40		
10	C		N40	USE	SUB	50	OVER	5	0				
11	C		N40	OVER	MULT	.04	BILL						
12	C		N40	BILL	ADD	2.50	BILL						
13	C		N40		GOTO	END							
14	C			USE	COMP	300					50		
15	C		N50	USE	SUB	100	OVER						
16	C		N50	OVER	MULT	.03	BILL						
17	C		N50	BILL	ADD	4.50	BILL						
18	C		N50		GOTO	END							
19	C			USE	SUB	300	OVER						
20	C			OVER	MULT	.025	BILL			H			
	C			BILL	ADD	10.50	BILL						
	C			END	TAG								
	C												

645

BEST COPY AVAILABLE

BEST COPY AVAILABLE

RPG II BUSINESS APPLICATION PROGRAMS

UNIT XV

Assignment Sheet 3 (con't)

Note: Your field names may vary

4. The Solution:

File Description Specifications

F Line	Form Type	Filename	File Type	File Designation	End of File	Sequence	File Format	Block Length	Record Length	L/R	Length of Key Field or of Record Address Field	Record Address Type	Type of File Organization or Additional Area	Overflow Indicator	Key Field Starting Location	Extension Code E/L	Device	Symbolic Device	Labels S/N/E/M	Name of Label Exit
02	F	ELECTRIC	I	P					20								DISK			
03	F	PRINT	O						132					OF			PRINTER			
04	F																			

IBM International Business Machines Corporation RPG INPUT SPECIFICATIONS

Program		Keying Instruction	Graphic
Programmer	Date		Key
Card Electro Number			

I Line	Form Type	Filename or Record Name	Sequence	Number (1/N), E	Option (O), U, S	Record Identifying Indicator, **, or DS	Record Identification Codes 1	Record Identification Codes 2	Record Identification Codes 3	Stacker Select	P/B/L/R	Field Location From	Field Location To	Decimal Positions	RPG Field Name	Control Level (L1-L9)
01	I	ELECTRIC	NS			01										
02	I											2	3	0	CUSNO	
03	I											4	8	0	NEW	
04	I											9	13	0	LAST	
05	I															
06	I															
07	I															
08	I															
09	I															
10	I															
11	I															
12	I															
13	I															

Assignment Sheet 4 (con't)

IBM International Business Machines Corporation

RPG OUTPUT SPECIFICATIONS

Program		Keying Instruction	Graphic	Card Electro Number
Programmer	Date		Key	

Commas	Zero Balances to Print	No Sign	CR	-
Yes	Yes	1	A	J
Yes	No	2	B	K
No	Yes	3	C	L
No	No	4	D	M

X = Remove Plus Sign
Y = Date Field Edit
Z = Zero Suppress
5-9 = User Defined

Line	Form Type	Filename or Record Name	AND/OR	Type (H/D/T/E)	Space Before	Space After	Skip Before	Skip After	Output Indicators	Field Name or EXCPT Name	Edit Codes	B/A/C/1-9=R	End Position in Output Record	P/B/L/R	Constant or Edit Word
01	O	PRINT		H		2			1P						
02	O		OR						OF						
03	O												27		'CUSTOMER ELECTRIC BILLS'
04	O			H		1			1P						
05	O		OR						OF						
06	O												27		'CUSTOMER NO. USAGE'
07	O												37		'BILL'
08	O			D		1			01						
09	O									CUSNO			7		
10	O									USE	Z		23		
11	O									BILL	1		38		
12	O														
13	O														
14	O														
15	O														
16	O														
17	O														
18	O														
19	O														
20	O														
	O														

647

BEST COPY AVAILABLE

Assignment Sheet 4

File Description Specifications

F

Line	Form Type	Filename	File Type	File Designation	End of File	Sequence	File Format	Block Length	Record Length	L/R	Length of Key Field or of Record Address Field	Record Address Type	Type of File Organization or Additional Area	Overflow Indicator	Key Field Starting Location	Extension Code E/L	Device	Symbolic Device	Labels S/N/E/M	Name of Label Exit
02	F	PAYIND	I	P					100		8	A	I		1		DISK			
03	F	PAYSEQD	O						100								DISK			
04	F	EXCPT	O						132								PRINTER			
05	F																			

IBM International Business Machines Corporation

RPG INPUT SPECIFICATIONS

Program

Programmer

Date

Keying Instruction: Graphic / Key

Card Electro Number

I

Line	Form Type	Filename or Record Name	Sequence	Number (1/N), E	Option (O), U, S	Record Identifying Indicator, **, or DS	Record Identification Codes 1	Record Identification Codes 2	Record Identification Codes 3	Stacker Select	P/B/L/R	Field Location From	Field Location To	Decimal Positions	RPG Field Name
01	I	PAYIND	NS			01									
02	I											1	100		RECORD
03	I											1	3	0	DEPT
04	I											4	8	0	NUMBER
05	I											9	17	0	SOCSEC
06	I											18	22	2	GROSS
07	I											23	27	2	FIT
08	I											28	32	2	SSTAX
09	I											53	56	2	DED1
10	I											57	60	2	DED2
11	I											61	64	2	DED3
12	I											65	68	2	DED4
13	I														

BEST COPY AVAILABLE

Assignment Sheet 4 (con't)

IBM International Business Machines Corporation

RPG CALCULATION SPECIFICATIONS

Program
Programmer
Date
Keying Instruction: Graphic / Key
Card Electro Number

C

Line	Form Type	Control Level (L0-L9, LR, SR, AN/OR)	Indicators	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Half Adjust (H)	Resulting Indicators High (Plus, 1 > 2)	Resulting Indicators Low (Minus, 1 < 2)	Resulting Indicators Equal (Zero, 1 = 2)
01	C			GROSS	SUB	FIT	NET						
02	C			NET	SUB	SSTAX	NET						
03	C			NET	SUB	DED1	NET					21	21
04	C		21	NET	ADD	DED1	NET						
05	C		21		EXCPT								
06	C		21		GOTO	END							
07	C			NET	SUB	DED2	NET					22	22
08	C		22	NET	ADD	DED2	NET						
09	C		22		EXCPT								
10	C		22		GOTO	END							
11	C			NET	SUB	DED3	NET					23	23
12	C		23	NET	ADD	DED3	NET						
13	C		23		EXCPT								
14	C		23		GOTO	END							
15	C			NET	SUB	DED4	NET					24	24
16	C		24	NET	ADD	DED4	NET						
17	C		24		EXCPT								
18	C			END	TAG								
19	C												
20	C												
001	C				SETOF						22	23	24
	C												
	C												
	C												
	C												

649

BEST COPY AVAILABLE

Assignment Sheet 4 (con't)

IBM International Business Machines Corporation

RPG OUTPUT SPECIFICATIONS

Line	Form Type	Filename or Record Name	Type (H/D/T/E)	Space	Skip	Output Indicators	Field Name or EXCPT Name	Edit Codes	End Position in Output Record	Constant or Edit Word
01	O	PAYSEQD	D			01				
02	O						RECORD		100	
03	O	EXCPT	H	3	06	1P				
04	O		OR			OF				
05	O						UDATE	Y	18	
06	O								66	'PAYROLL DEDUCTION'
07	O								77	'EXCESSES'
08	O								114	'PAGE'
09	O						PAGE	Z	118	
10	O		E	2		21				
11	O		OR			22				
12	O		OR			23				
13	O		OR			24				
14	O						DEPT	Z	3	
15	O						NUMBER	Z	10	
16	O						SOCSEC		24	'0 - - '
17	O						GROSS	1	32	
18	O						FIT	1	40	
19	O						SSTAX	1	48	
20	O						DED1	1	55	
	O						DED2	1	62	
	O						DED3	1	69	
	O						DED4	1	76	
	O								95	'ACTUAL NET PAY IS'
	O						NET	1	103	

IBM International Business Machines Corporation

RPG OUTPUT SPECIFICATIONS

Line	Form Type	Filename or Record Name	Type (H/D/T/E)	Space	Skip	Output Indicators	Field Name or EXCPT Name	Edit Codes	End Position in Output Record	Constant or Edit Word
01	O					21			124	'NO DEDUCTIONS TAKEN'
02	O					22			117	'INCLUDING D1'
03	O					23			121	'INCLUDING D1, D2'
04	O					24			125	'INCLUDING D1, D2, D3'
05	O									
06	O									

BEST COPY AVAILABLE

RPG II BUSINESS APPLICATION PROGRAMS

UNIT XV

JOB SHEET I: Execute Accounts Receivable Register on Computer

I. Equipment

Computer
Floppy Disk

II. Procedure

Directions: Using the RPG program written for AS I, key in, compile, and execute program on computer. Produce the required output as specified and compare with solution on corresponding assignment sheet.

Note: No alterations are needed to execute this program on an IBM System 34, 36, or 38. Minor modifications may be necessary for other makes of computer equipment. However, any system must have an RPG compiler to execute.

RPG II BUSINESS APPLICATION PROGRAMS

UNIT XV

JOB SHEET 2: Execute Master Subscriber File Program

I. Equipment

Computer
Floppy Disk

II. Procedure

Directions: Using the RPG program written for AS 2, key in, compile, and execute program on computer. Produce the required output as specified and compare with solution on corresponding assignment sheet.

Note: No alterations are needed to execute this program on an IBM System 34, 36, or 38. Minor modifications may be necessary for other makes of computer equipment. However, any system must have an RPG compiler to execute.

JOB SHEET 3: Execute Electric Bill Problem

I. Equipment

Computer
Floppy Disk

II. Procedure

Directions: Using the RPG program written for AS 3, key in, compile, and execute program on computer. Produce the required output as specified and compare with solution on corresponding assignment sheet.

Note: No alterations are needed to execute this program on an IBM System 34, 36, or 38. Minor modifications may be necessary for other makes of computer equipment. However, any system must have an RPG compiler to execute.

RPG II BUSINESS APPLICATION PROGRAMS

UNIT XV

JOB SHEET 4: Execute Payroll Calculation Program

I. Equipment

Computer
Floppy Disk

II. Procedure

Directions: Using the RPG program written for AS 4, key in, compile, and execute program on computer. Produce the required output as specified and compare with solution on corresponding assignment sheet.

Note: No alterations are needed to execute this program on an IBM System 34, 36, or 38. Minor modifications may be necessary for other makes of computer equipment. However, any system must have an RPG compiler to execute.

654

RPG II BUSINESS APPLICATION PROGRAMS

UNIT XV

EXAMINATION

1.1. Define what a company's accounts receivable is.

1.2. List 3 purposes of accounts receivable.

a.

b.

c.

1.3. List 3 of the 5 basic characteristics of an accounts receivable transaction register.

a.

b.

c.

2.1. Define file maintenance.

2.2. List 3 functions of file maintenance.

a.

b.

c.

3. List 6 objectives of a computerized billing system.

a.

b.

c.

d.

e.

f.

4.1. Identify benefits of computerized payroll.

a.

b.

c.

4.2. Name 6 typical deductions found in a payroll application.

a.

b.

c.

d.

e.

f.

RPG II BUSINESS APPLICATION PROGRAMS

UNIT XV

ANSWERS TO EXAMINATION

1.1. Amount of money owed to a company by its customers for merchandise sold or services rendered.

1.2. a. Facilitate collection of money owed by customers
b. Minimize bad debts losses
c. Maintain customer goodwill - accurate recordkeeping

1.3. (Any 3 of the following characteristics)
a. Runs daily
b. Lists of all transactions concerning A/R
c. Confirms invoices and receipts
d. Uses invoice summary file
e. Reports sales amount vs. income amount

2.1. File maintenance - performing functions to keep a file current for daily processing.

2.2. a. Adding records
b. Updating records
c. Tagging/flagging records for deletion

3.1. a. Record and acknowledge customer electric bills
b. Compute prices, discounts, and totals (taxes, if applicable)
c. Automatically produce timely and accurate customer bills through cyclic billing during the month
d. Provide necessary management reports and inquiry capabilities for their customers; i.e., past-due customer reports and individual customer states
e. Reduce paper flow throughout office
f. Reduce clerical time and errors from beginning to end of billing cycle

4.1. a. Analyze and control company payroll costs
b. Allow for clerical savings
c. Facilitate audit procedures

4.2. a. Federal tax
b. FICA tax
c. Savings bonds
d. Life insurance
e. Hospitalization
f. State tax

RPG II TOPICS: TABLES AND ARRAYS

UNIT XVI

UNIT OBJECTIVES

After studying this unit, students should be able to describe tables and arrays, use them in calculations, and state the difference that exists between tables and arrays as used in the RPG II language.

SPECIFIC OBJECTIVES

After completion of this unit, students should be able to:

1. Define a Table and its 5 specific (basic) rules.

2. Explain what compile-timetables are and their relationship to the Extension Sheet.

3. Describe Pre-execution-time Tables and necessary File Description Sheet Coding.

4.1 Explain what the LOKUP (table look-up) operation accomplishes.

4.2 Search a table using the LOKUP operation on the calculation sheet.

5.1 Recognize similarities and differences between Tables and Arrays.

5.2 List coding sheets and describe necessary operations to print out the contents of a table or array at the end of a job.

5.3 Print out table or array contents at the end of a job.

RPG II TOPICS: TABLES AND ARRAYS

UNIT XVI

SUGGESTED ACTIVITIES

I. Instructor

   A. Make transparencies.

   B. Provide students with objectives and discuss.

   C. Provide students with the following coding sheets: Extension and Line Counter, Control and File Description, Input, Calculations, and Output.

   D. Provide students with information and assignment sheets and discuss.

   E. Give examination.

II. Student

   A. Read objectives.

   B. Study information sheets as presented and assigned by instructor.

   C. Compile assignment sheets.

      1. Each student codes a simple table and array exercise from assignment sheets onto coding sheets.

      2. Instructor checks work for accuracy and completeness.

      3. Student is given further orientation by instructor on tables and arrays. (Review information sheets.)

      4. Student then codes additional assignment sheets problems. (Complete job sheets.)

INSTRUCTIONAL MATERIALS

I. Unit Content

   A. Objectives

   B. Information sheets

   C. Handouts

      1. HO 1: Table Look-up Example

      2. HO 2: Calculations and GOTO Operation

3. HO 3: Two Related Tables
4. HO 4: Sample Table Problem and Coded Solution
5. HO 5: Example of an Array
6. HO 6: Using XFOOT and MOVEA
7. HO 7: Indexing with Arrays
8. HO 8: Using a Field as the Index

D. Transparency masters

1. TM 1: Problem System Flowchart
2. TM 2: Types of Tables and Arrays
3. TM 3: Table and Array Data Operations
4. TM 4: Examples of XFOOT and MOVEA

E. Assignment sheets

1. AS 1: Sample Assignment Sheet Program
2. AS 2: Coding Extension Sheets I
3. AS 3: Coding Extension Sheets II
4. AS 4: Describing a Pre-execution-time Table
5. AS 5: Searching a Table
6. AS 6: Finding a Value in a Table
7. AS 7: Array Problem and Questions

F. Answers to assignment sheets

G. Examination

H. Answers to examination

II. References

A. _IBM System 34/36/38 RPG II Reference Manual_. Information Development. Rochester, MN: IBM Corp., 1983.

B. Shelly, Gary B. and Thomas J. Cashman. _Computer Programming RPG II_. Fullerton, CA: Anaheim Publishing Company, 1978.

C. _IBM RPG II Self Study_. (Inquire at Guided Learning Center of your local IBM branch office for availability of classes and textbooks.)

RPG II TOPICS: TABLES AND ARRAYS

UNIT XVI

NOTES TO THE INSTRUCTOR

A. There are data processing problems in which it is convenient to store information in tables so that a desirable piece of data, such as a shipping cost can be found and applied during calculations.

Here are some instances where it might be used:

1. When an item number is known, look up its description.
2. When a length is known in inches, look up its equivalent in centimeters.
3. When a temperature is given in degrees Fahrenheit, find its equivalent in degrees Centigrade.

B. In order to describe tables and arrays to students, this unit will use an Extension Specification Sheet. In addition to this guide, your students will need a few sheets of each of the following forms:

Extension and Line Counter

Control and File Description

Input

Calculation

Output

C. A somewhat different approach will be used in this unit to introduce the topic of tables. We'll start by stating a problem and then show a coded solution. A discussion will follow about the table data entries.

D. New coding entries encountered in this chapter include:

File Description

16	File Designation (T for table file)
39	Extension Code (E)

Extension

11 - 18	From Filename (pre-execution-time tables)
19 - 26	To Filename (to write out table values at end of job)
27 - 32	Table or Array Name

33 - 35	Number of Entries Per Record
36 - 39	Number of Entries Per Table or Array
40 - 42	Length of Entry
44	Decimal Position
45	Sequence (Ascending)
46 - 51	Table or Array Name (Alternating Format)
52 - 54	Length of Entry

E. Within this unit there are a number of assignment sheets dealing with tables and arrays. Job sheets were intentionally left out due to additional set-up time required of an instructor. Should the instructor wish to create necessary tables and arrays for corresponding programs, then these jobs could certainly be run on the computer.

RPG II TOPICS: TABLES AND ARRAYS

UNIT XVI

INFORMATION SHEET

I. Tables

  A. Definition - a table is an arrangement of data items having like characteristics. Each data item in a table is known in RPG II as a table entry or element.

  B. Sample Problems (TM 1)

    1. Program requirements (AS 1)

    2. Program solution

    3. Program explanation

  C. Specific Rules for Tables (AS 2 and 3)

    1. Every table entry must be the same length.

    2. Every table entry must be of the same type, either all numeric or all alphameric.

    3. Every entry in a numeric table must have the same number of decimal positions.

    4. Every table name must start with TAB and may include additional letters or numbers in any combination.

    5. Each table used in a program is defined on the extension specifications sheet.

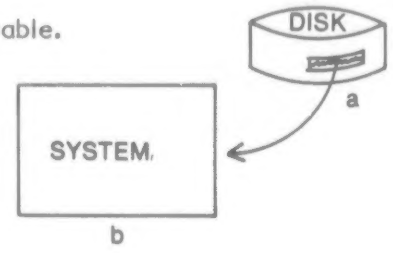
II. Compile-time Tables (TM 2)

  A. Description - The tables described up to this point are known as "compile-time tables." This is the actual table data that are included as a part of a RPG II source program.

  B. Procedure when using compile-time tables

    1. Load the program that includes the table.

    2. Start the job.



C. Order in which coding will be keyed in or read in from disk for compilation of program using 2 compile-time tables

1. File Description specifications
2. Extension specifications
3. Input specifications
4. Calculations specifications
5. Output specifications
6. A special record containing **b̸ in 1-3 (b̸ means blank)
7. All of the actual table data records for the first table as defined on the Extension sheet
8. A second special record (**b̸ in 1-3)
9. Actual table data records for the second table defined on the Extension sheet

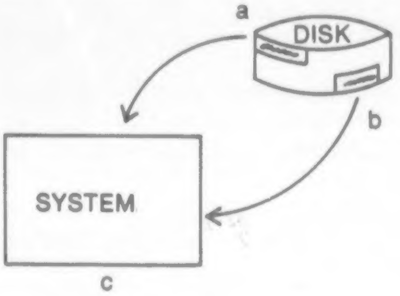
D. During the process of compilation, these tables are stored with the program for future use.

E. Purpose of the Extension sheet - is used to describe every table used in a program. The **b̸ records are included in a source program whenever compile-time tables are required.

III. Pre-execution-time tables

A. Description - Pre-execution-time tables are created some time before they are needed and they are stored on the disk as a file. A table of this type is also defined in the same manner on the Extension sheet but must include one more specification. Enter a filename in positions 11-18 which is known as the "From Filename." When this is the case, a compiled object program is loaded to run a job that is followed immediately by all pre-execution-time tables defined as files for it. Then the job starts to run.

1. Load the program, that
2. Loads the table or array, then
3. Start the job.



B. Rule: When pre-execution-time tables are used in a program, each one must be described as an input table file on the File Description sheet and must include these entries.

1. A T in position 16 (code for a table file)

2. An E in position 39 (code for extension entries needed) to alert the RPG II program compiler to tie together the File Description entry and the Extension entry having the same filename when used in the object program.

(NOTE: This rule applies only to pre-execution-time tables.)

IV. Table look-up in a program

A. Table look-up is this kind of activity. (TM 3)

1. A "search" field value is compared against the first table entry.

2. If the table entry is what is being searched for, the table value is made available for processing and output.

3. If the first table entry is not what is being searched for, the program compares the search value against the next table value. Again, the question (in 2 above) is raised.

4. If no acceptable table entry is found during the search and the entire table has been searched, the next step following the operation is begun.

B. Steps to code up the table look-up activity (HO 1)

1. Step 1 - describes the tables to be used by making entries on the Extension sheet and on the File Description sheet if needed.

2. Step 2 - describes an input field (or a constant) that contains (or is) the searching value.

3. Step 3 - describes the LOKUP (table look-up) operation including:

   a. The "search" field or constant as Factor 1

   b. LOKUP as the operation

   c. The table to be searched as Factor 2

   d. Another table which contains corresponding values as the Result Field (optional)

   e. At least 1 Resulting Indicator (in positions 54-59)

C. Search Table Entries AS 5, TM 3)

1. Reason for doing a table look-up operation - to use the value found in the corresponding table (the one in the Result Field) for additional calculations

2. Reason to bypass all further calculations. If no search table entry (the Factor 2 table) satisfies the problem requirement, such as "search for an equal value," then all further calculations may be bypassed or computer run may be stopped.

3. Reason to stop computer run - if table data, the search field data, or the constant is wrong, then the operator should probably rerun the job after corrections have been made. Otherwise, if the job continues, all output will be garbage.

D. GOTO Operation (HO 2)

1. How to bypass calculation steps - simply include a GOTO operation.

2. How to stop the computer during a run - system operators press a stop key on the machine. The programmer uses a special "halt" indicator to do the same thing.

3. Rule: When a halt indicator is turned on, processing continues for all remaining steps, including output for that record, and then the computer stops.

E. High, Low, and Equal - testing the look-up activity (AS 6)

1. Introduction - the look-up activity may be tested for 3 conditions: High, Low, and Equal. The controlling specification for this operation is the table named as Factor 2 on the Coding sheet. Examine the situation where it is useful to have an indicator assigned to High (54-55) or Low (56-57).

2. Rules to follow when looking for the above condition

   a. If a table is to be searched for a "high" condition, table values must be in ascending sequence.

   b. If a table is to be searched for a "low" condition, table values must be in descending sequence.

   (NOTE: Look at an Extension sheet. An entry is made in position 45 whenever tables are to be searched for either a "high" or a "low" condition. Specify code A for ascending or code D for descending table values.

c. At least one indicator must be assigned but no more than two indicators may be entered for a look-up operation. In RPG coding, this provides for five acceptable combinations.

1) High (54-55): search an ascending table until a value in it is higher than the search value.

2) Low (56-57): search a descending table until a value in it is lower than a search value.

3) Equal (58-59): search a table until an equal is found. This table may be in any sequence or it can be randomly arranged.

4) High (54-55) and Equal (58-59): search an ascending table until a value in it is either equal (tested first) or a high condition exists.

5) Low (55-56) and Equal (58-59): search a descending table until a value in it is either equal (tested first) or a low conditon exists.

d. When 2 conditions are to be tested, the same indicator may be assigned to both places.

(Note: Refer to the RPG II Reference Manual which contains additional rules and examples about tables. Ignore information about arrays as they will be covered later.)

F. "Related tables can be described separately or in alternating format." (HO 3, 4, and 5)

1. Tables that can be described separately or in an alternating format

2. Instead of describing two tables whose values are related using two separate lines on an Extension sheet, one line is used to describe a combination table that alternatingly contains a value for one table followed by a value for the other table and so on.

3. When the alternating format is used, fill in entries in positions 27-45 to describe the table values in the first part of each group and then specify a second table name and information using positions 46-57 for table values in the second part.

V. Arrays

A. Similarities between tables and arrays

1. An array is a table. That is, an array is an arrangement of data items having like characteristics.

2. Each array used in a program is described on the Extension specifications sheet.

3. There are "compile-time" arrays and "pre-execution-time" arrays. Both are used and specified in the same manner as are tables of these same two types.

4. A look-up operation (LOKUP) may be specified for calculations. There are five acceptable combinations for testing the results of an array look-up: High, High and Equal, Low, Low and Equal, or Equal only.

5. Related arrays can be described separately or in alternating format.

B. Differences between tables and arrays

1. Only table names may begin with TAB. Array names can be any other field name you wish to make up.

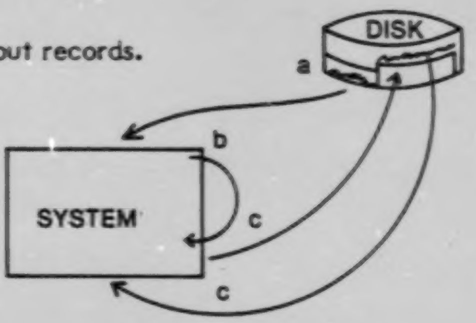
   Rule: An array name refers to the entire array.

2. There is a special kind of array called an "execution-time" array. In addition to describing this kind of array on an Extension sheet, you also describe it on an input record on an Input sheet. (HO 5)

   a. Load the program, to

   b. Start the job, that

   c. Reads array data as input records.



3. In addition to the use of LOKUP to perform a look-up operation on either table or array data, two special operations to process array data are available.

   a. XFOOT (crossfoot) causes the value in each entry of a numeric array to be added together and stored in the Result Field.

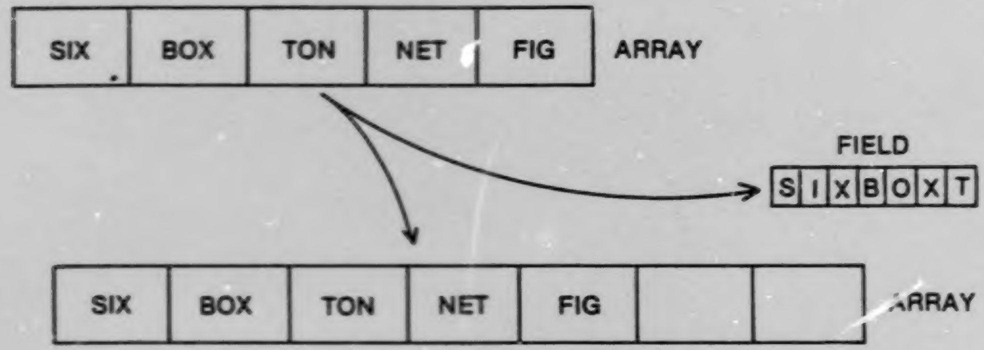
0	1	0	0	6	4	2	0	1

0 + 1 + 0 + 0 + 6 + 4 + 2 + 0 + 1 = 14

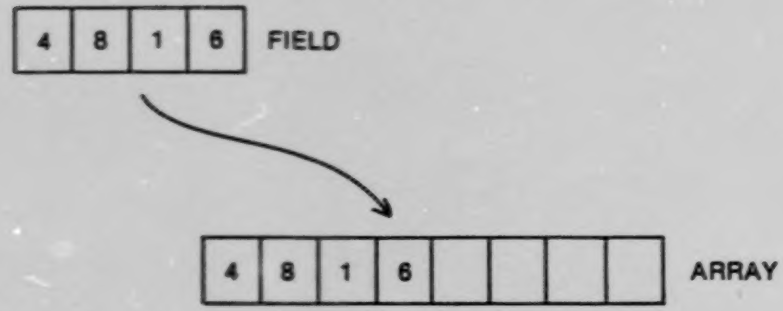
b. MOVEA (move array) causes characters from the leftmost positions of Factor 2 to be moved to the leftmost positions of the Result Field. Using this operation enables you to do one of three things:

1) Move data in one array to replace data in another array.

2) Move part or all of the data in an array into a field.


SIX
BOX
TON
NET
FIG
ARRAY
FIELD
S I X B O X T
SIX
BOX
TON
NET
FIG
ARRAY

3) Move part or all of the data in a field into an array.


4
8
1
6
FIELD
4
8
1
6
ARRAY

Rule: Movement of data stops when the end of the shorter length item (the array or the field) is reached. (HO 6, TM 4)

4. When tables are edited on the Output sheet, a table entry is edited just as a numeric field is edited. An array entry may be edited in the same manner. Specify an edit code in position 38 on Output and the array entry will be edited as desired.

5. An array may be edited in its entirety. If you enter an edit code in 38 and name the array, RPG II will cause the array to be edited so that each entry in the array will include the desired punctuation, and 2 spaces will be included to separate each entry from the other when printed. Rule: Be sure to plan for enough space on the print line to include all punctuation characters as well as the 2 spaces between items.

6. Entire tables and arrays on an Extension sheet have been described. Search either a table or an array to find a particular item using the LOKUP operation.

7. RPG II provides a unique facility for the referencing and use of particular array entries through the use of a scheme known as "indexing." In this case: (HO 7)

    a. Use an array name of 1, 2, or 3 characteristics.

    b. Include an "index" constant or field to refer to the desired entry.

C. Using a field as the Index in an Array (HO 8)

1. The array AR has 5 entries, each containing 1 numeric character.

2. On the fourth calculation step, refer to a particular entry of array AR. The particular entry is determined by the value in the index field Y.

3. The first time through, Y is given the value 1 as a result of the Z-ADD operation. This means to use the first entry of array AR on the first try for the COMP step.

4. The value of Y is incremented (increased) by 1 each time the loop is completed. When either the value in an entry of the array is 7 or the value of the index is 5, this activity is finished.

D. Tables and Arrays at LR (Last Record)

1. Rule: Entire tables and arrays can be written under control of RPG II at the end of a job when the last record indicator (LR) is on.

2. Describing Tables or Array

    a. Describe an Output file on the File Description sheet to hold those tables and/or arrays.

    b. Include the filename as an entry on the Extension sheet in positions 19-26 for every table and array to be written out. Positions 19-26 are titled "To Filename."

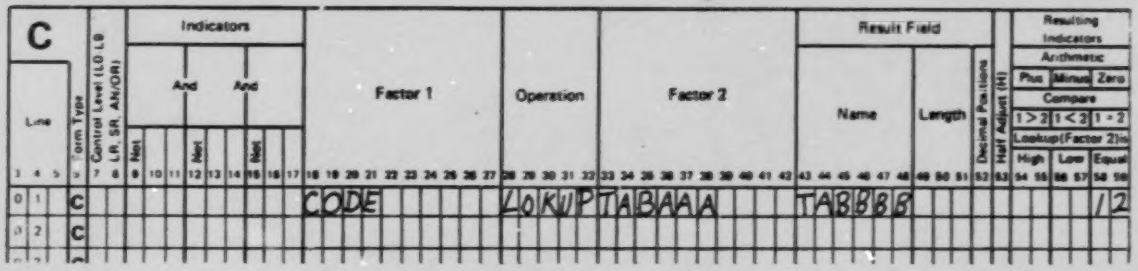
    c. Describe a total-time output record (Type code T in 15) under control of the last record indicator (LR). Include the name of the array or array entries or table to be produced as output in positions 32-37 (Field Name).

RPG II TOPICS: TABLES AND ARRAYS

UNIT XVI

HANDOUT I: Table Look-up Example

Here is an example of table look-up as coded on a Calculation sheet.



Line (3-5)	Form Type (6)	Control Level (L0-L9, LR, SR, AN/OR) (7-8)	Indicators (9-17)	Factor 1 (18-27)	Operation (28-32)	Factor 2 (33-42)	Result Field Name (43-48)	Result Field Length (49-51)	Decimal Positions (52)	Half Adjust (H) (53)	Resulting Indicators High (54-55)	Low (56-57)	Equal (58-59)
01	C			CODE	LOKUP	TABAAA	TABBBB						12
02	C												

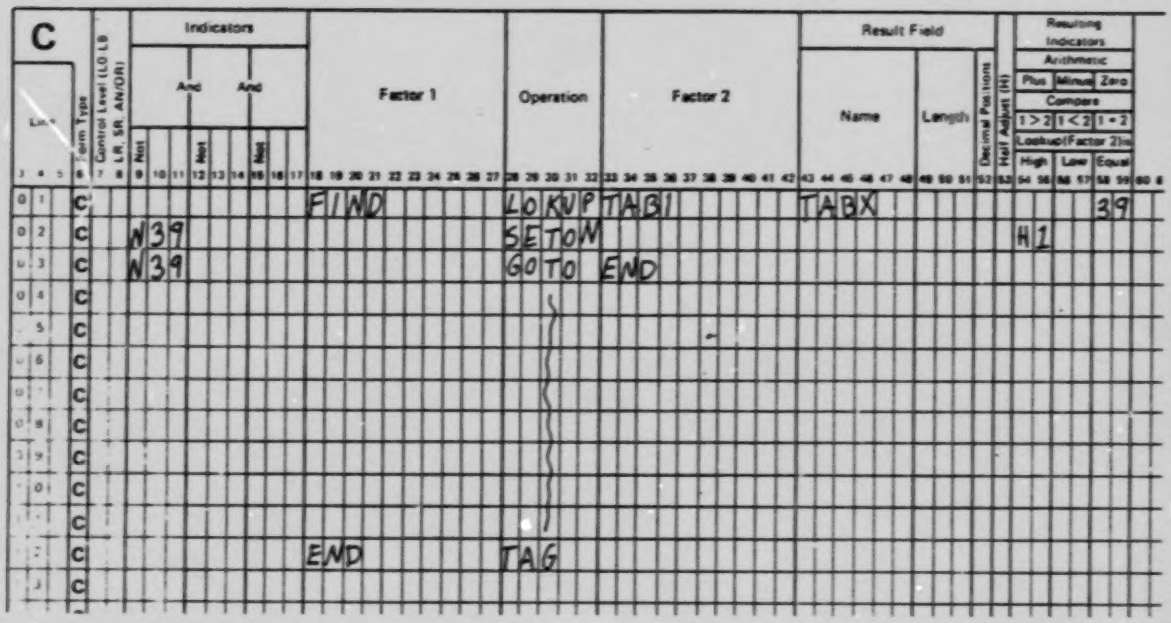
This description means "search the table named TABAAA for a value equal to (indicator 12 is in positions 58-59) the value found in the search field named CODE. When you find an equal, also locate a corresponding value in the table named TABBB."

RPG II TOPICS: TABLES AND ARRAYS

UNIT XVI

HANDOUT 2: Calculations and GOTO Operation

Consider what happens in the example below when the entire table (TABI) has been searched and no value equal to the value in FIND can be located. First, indicator 39 is not turned on. Second, it tells the computer to "set on" a special halt indicator, HI. Third, it also directs the program to bypass all further calculations.



```
01   C           FIND      LOKUPTABI      TABX           39
02   C  N39                SETON                    H1
03   C  N39                GOTO END
04   C
05   C
06   C
07   C
08   C
09   C
10   C
11   C
12   C           END       TAG
13   C
```

Using the GOTO assures the programmer that all further processing will be bypassed, but the machine does not stop running! By including the SETON instruction and any one of the halt indicators (HI-H9), the machine will stop after all processing and output for that record have been completed. The operator then becomes aware of the fact that something has gone wrong in the program. The programmer must provide a set of instructions to the operator for this situation so the job may be either be completed with the error noted or the job removed until the error can be corrected.

After the operator takes action, including restarting the machine, the halt indicator is automatically turned off.

BEST COPY AVAILABLE

RPG II TOPICS: TABLES AND ARRAYS

UNIT XVI

HANDOUT 3: Two Related Tables

Related tables can be described separately or in alternating format. This means that instead of describing two tables whose values are related by using two separate lines on an Extension sheet, one line is used to describe a combination table that alternatingly contains a value for one table followed by a value for the other table, and so on.

As Two Related Tables:

Extension Specifications

Line	Form Type 6	Record Sequence of the Chaining File 7-8	Number of the Chaining Field 9-10	From Filename 11-18	To Filename 19-26	Table or Array Name 27-32	Number of Entries Per Record 33-35	Number of Entries Per Table or Array 36-39	Length of Entry 40-42	P/B/L/R 43	Decimal Positions 44	Sequence (A/D) 45	Table or Array Name (Alternating Format) 46-51	Length of Entry 52-54	P/B/L/R 55	Decimal Positions 56	Sequence (A/D) 57
	E					TAB1	7	7	1		0	A					
	E					TAB2	7	7	9								
	E																

In Alternating Format:

When the alternating format is used, fill in entries in positions 27-45 to describe the table values in the first part of each group and then specify a second table name and information using positions 46-57 for table values in the second part. Here is an example.

Extension Specifications

Line	Form Type 6	Record Sequence of the Chaining File 7-8	Number of the Chaining Field 9-10	From Filename 11-18	To Filename 19-26	Table or Array Name 27-32	Number of Entries Per Record 33-35	Number of Entries Per Table or Array 36-39	Length of Entry 40-42	P/B/L/R 43	Decimal Positions 44	Sequence (A/D) 45	Table or Array Name (Alternating Format) 46-51	Length of Entry 52-54	P/B/L/R 55	Decimal Positions 56	Sequence (A/D) 57
	E					TAB1	7	7	1		0	A	TAB2	9			
	E																

BEST COPY AVAILABLE

RPG II TOPICS: TABLES AND ARRAYS

UNIT XVI

HANDOUT 4: Sample Table Problem and Coded Solution

Since RPG II can use tables described separately or in alternating format, you need to know how your company wishes to arrange table data before creating table data values and then describe them as created in your problem solutions.

To complete the presentation on Tables, a coded solution to an example in which we wish to look up phone numbers is included. A list is then printed.

Problem:

NAME TABLE	NUMBER TABLE
Baker Diane	286-6424
Boyd Alan	289-2933
Brockmeier Ric	938-7515
Lamkin Tim	935-8381
Lock Joey	288-7587
Schnare Sharon	938-3932
Whipkins Glen Ray	288-4663

File Description Specifications

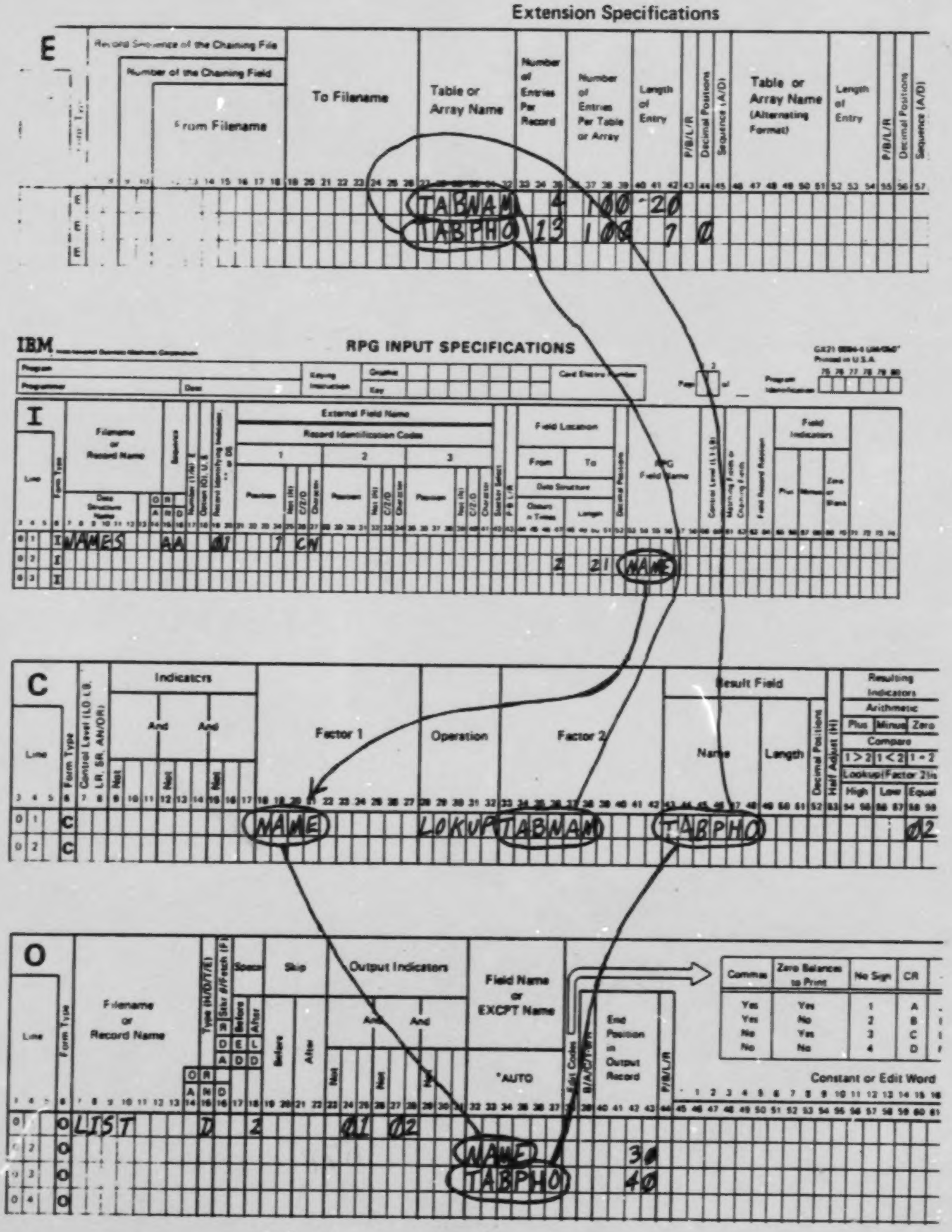
Line (3-5)	Form Type (6)	Filename (7-14)	File Type I/O/U/C/D (15)	File Designation P/S/C/R/T/D/F (16)	End of File E (17)	Sequence A/D (18)	File Format F/V/S/M/D/E (19)	Block Length (20-23)	Record Length (24-27)	Mode of Processing L/R (28)	Length of Key Field or of Record Address Field (29-30)	Record Address Type A/P/I/K (31)	Type of File Organization or Additional Area I/X/D/T/R/ or 2 (32)	Overflow Indicator (33-34)	Key Field Starting Location (35-38)	Extension Code E/L (39)	Device (40-46)	Symbolic Device (47-52)	Labels S/N/E/M (53)	Name of Label Exit (54-59)
02	F	NAMES	I	P				25	23								CONSOLE			
03	F	LIST	O						132								PRINTER			
04	F																			

External Record Name

674

BEST COPY AVAILABLE

Coded Problem Solution (con't):


Extension Specifications
E
Record Sequence of the Chaining File
Number of the Chaining Field
From Filename
To Filename
Table or Array Name
Number of Entries Per Record
Number of Entries Per Table or Array
Length of Entry
Table or Array Name (Alternating Format)
IBM
RPG INPUT SPECIFICATIONS
I
Filename or Record Name
Field Location
RPG Field Name
Field Indicators
C
Indicators
Factor 1
Operation
Factor 2
Result Field
Name
Length
Resulting Indicators
O
Filename or Record Name
Space
Skip
Output Indicators
Field Name or EXCPT Name
End Position in Output Record
Constant or Edit Word

```
     E                    TABNAM   4 100 20
     E                    TABPHO  13 100  7 0
     E

01   I NAMES     AA  01     1 CN
02   I                               2  21 NAME
03   I

01   C           NAME      LOKUPTABNAM     TABPHO          02
02   C

01   O LIST    D  2      01 02
02   O                        NAME      30
03   O                        TABPHO    40
04   O
```

BEST COPY AVAILABLE

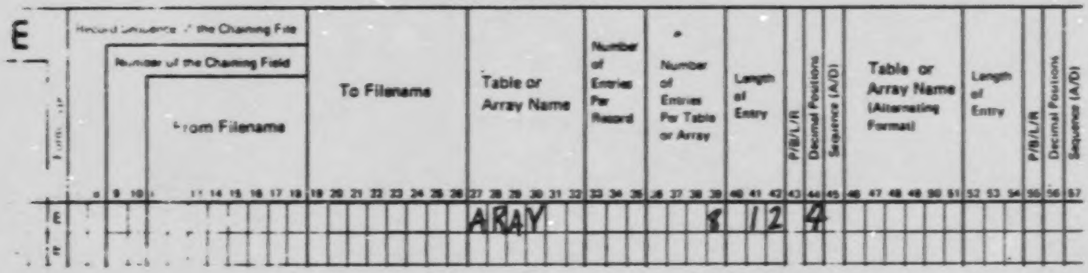
RPG II TOPICS: TABLES AND ARRAYS

UNIT XVI

HANDOUT 5: Example of an Array

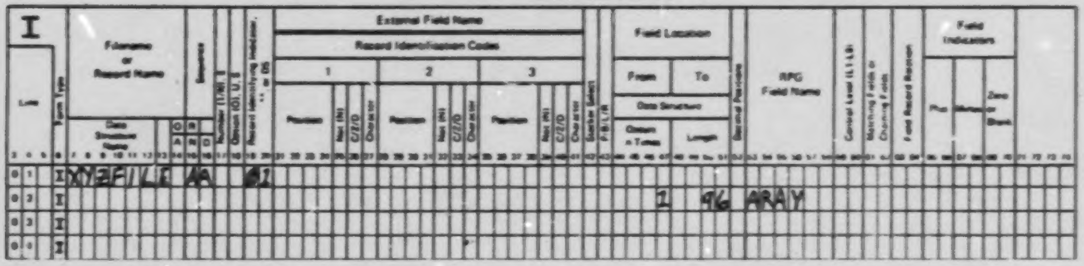
An example of an "Execution-time" array is shown below.

Extension Specifications



Form Type	Record Sequence of the Chaining File	Number of the Chaining Field	From Filename	To Filename	Table or Array Name	Number of Entries Per Record	Number of Entries Per Table or Array	Length of Entry	P/B/L/R	Decimal Positions	Sequence (A/D)	Table or Array Name (Alternating Format)	Length of Entry	P/B/L/R	Decimal Positions	Sequence (A/D)
E					ARAY		8	12		4						
E																

RPG INPUT SPECIFICATIONS



Line	Form Type	Filename or Record Name	Sequence	Record Identifying Indicator	Field Location From	Field Location To	RPG Field Name
01	I	XYZFILE	AA	01			
02	I				1	96	ARAY
03	I						
04	I						

When this kind of array is described on the Extension sheet, no entry is made in positions 33-35 (Number of Entries Per Record) because the input field or fields will provide the data later on. What is shown above is essentially this. First, a storage space (Extension specification) is described for an 8 entry array in which each entry will have 12 positions of numeric data including 4 decimal positions. Second, the storage space of 96 positions (8 x 12 = 96) will be filled with data as it is read in from an input record during the job.

676

BEST COPY AVAILABLE

RPG II TOPICS: TABLES AND ARRAYS

UNIT XVI

HANDOUT 6: Using XFOOT and MOVEA

In addition to the use of LOKUP to perform a look-up operation on either table or array data, two special operations, XFOOT and MOVEA, are available to process array data.

An example of XFOOT (crossfoot) which causes the value in each entry of a numeric array to be added together and stored in the Result Field is shown below.

C Line 3-5	Form Type 6	Control Level (L0-L9, LR, SR, AN/OR) 7-8	Indicators 9-17	Factor 1 18-27	Operation 28-32	Factor 2 33-42	Result Field Name 43-48	Result Field Length 49-51	Decimal Positions 52	Half Adjust (H) 53	Resulting Indicators 54-59
01	C				XFOOT	ARAY	SUM1	14	4		
02	C										

An example of MOVEA (move array) which causes characters from the leftmost positions of Factor 2 to be moved to the leftmost positions of the Result Field is shown below.

C Line 3-5	Form Type 6	Control Level (L0-L9, LR, SR, AN/OR) 7-8	Indicators 9-17	Factor 1 18-27	Operation 28-32	Factor 2 33-42	Result Field Name 43-48	Result Field Length 49-51	Decimal Positions 52	Half Adjust (H) 53	Resulting Indicators 54-59
01	C				MOVEA	ARR1A	ARRA2				
02	C										
03	C				MOVEA	ARA60	FIELDX	4			
04	C										
05	C				MOVEA	FIELDT	AR29				
06	C										

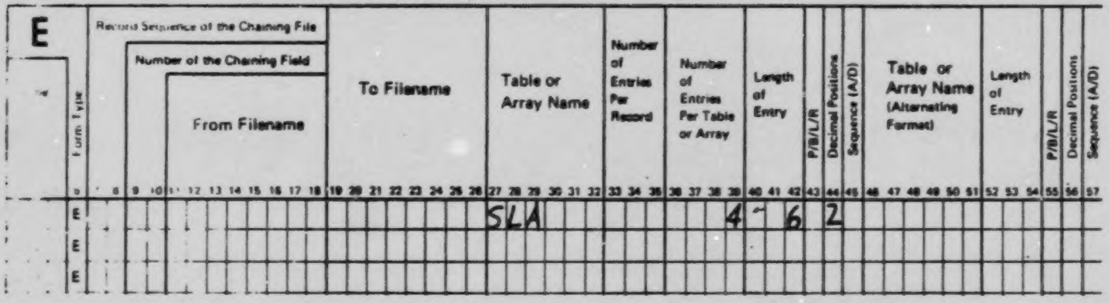
677

BEST COPY AVAILABLE

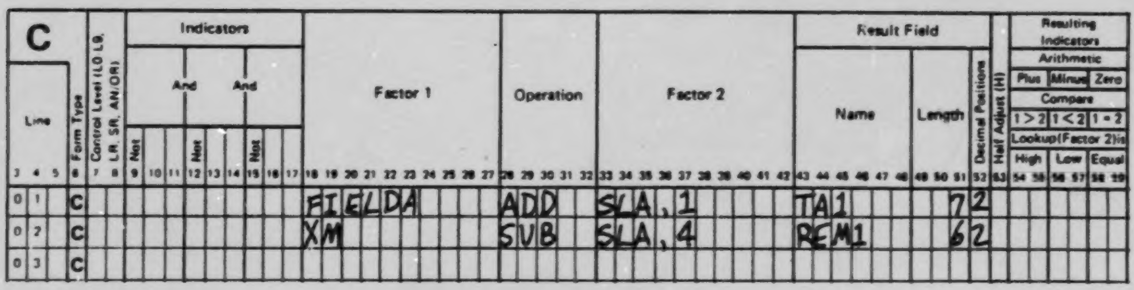
HANDOUT 7: Indexing with Arrays

RPG provides a unique facility for the referencing and use of particular array entries through the use of a scheme called "indexing." The illustrations below show this process.

Extension Specifications



Form Type	Record Sequence of the Chaining File	Number of the Chaining Field	From Filename	To Filename	Table or Array Name	Number of Entries Per Record	Number of Entries Per Table or Array	Length of Entry	P/B/L/R	Decimal Positions	Sequence (A/D)	Table or Array Name (Alternating Format)	Length of Entry	P/B/L/R	Decimal Positions	Sequence (A/D)
E					SLA		4	6		2						
E																
E																



Line	Form Type	Control Level (L0-L9, LR, SR, AN/OR)	Indicators	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Half Adjust (H)	Resulting Indicators
01	C			FIELDA	ADD	SLA,1	TA1	7	2		
02	C			XM	SUB	SLA,4	REM1	6	2		
03	C										

678 BEST COPY AVAILABLE

RPG II TOPICS: TABLES AND ARRAYS

UNIT XVI

HANDOUT 8: Using a Field as the Index

Tables and arrays are alike by definition, but arrays can be processed in a variety of ways not available for processing table data. We are not limited to using a constant as an index value. The following example shows the use of a field as the index. Take time to examine what is happening in this series of calculation steps.

Extension Specifications

E Line	Form Type	Record Sequence of the Chaining File	Number of the Chaining Field	From Filename	To Filename	Table or Array Name	Number of Entries Per Record	Number of Entries Per Table or Array	Length of Entry	P/B/L/R	Decimal Positions	Sequence (A/D)	Table or Array Name (Alternating Format)	Length of Entry	P/B/L/R	Decimal Positions	Sequence (A/D)
01	E					AR		5	1		0						
02	E																

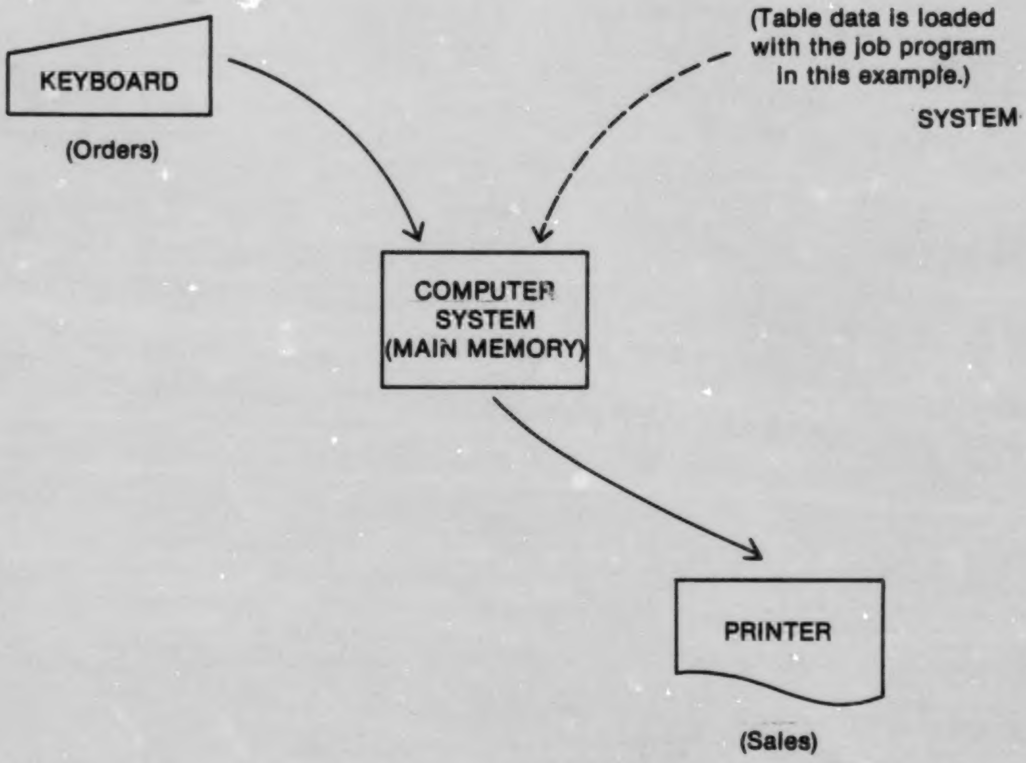
C Line	Form Type	Control Level (L0-L9, LR, SR, AN/OR)	Indicators	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Arithmetic Plus / Compare 1>2 / Lookup (Factor 2) is High	Minus / 1<2 / Low	Zero / 1=2 / Equal
01	C				SETOF						20	30	
02	C				Z-ADD	1	Y	1	0				
03	C			RET	TAG								
04	C			AR,Y	COMP	7							20
05	C		20		GOTO	END							
06	C			5	SUB	Y	TEST	1	0				30
07	C		30		GOTO	END							
08	C			Y	ADD	1	Y						
09	C				GOTO	RET							
10	C			END	TAG								
11	C												
12	C												

What you have just examined is the coding used to scan the contents of a field, one position at a time. In order to do this, the field was described as a 5-entry array and then a loop was programmed to examine the first, the second, the third, the fourth and the fifth entry for a particular character. In our example the scan was being made for the presence of the character 7.

679

BEST COPY AVAILABLE

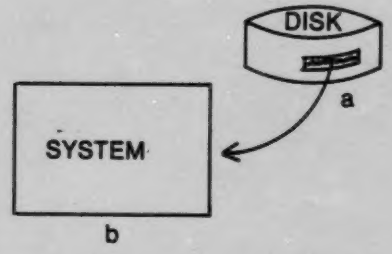
Problem System Flowchart


KEYBOARD
(Orders)
(Table data is loaded with the job program in this example.)
SYSTEM
COMPUTER SYSTEM (MAIN MEMORY)
PRINTER
(Sales)

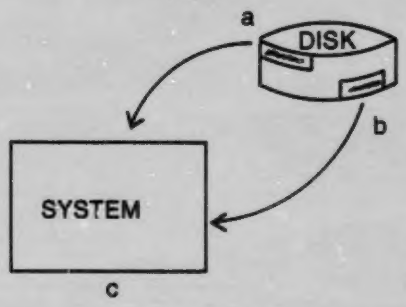
TM 2

Types of Tables and Arrays

Compile-time Table Or Array


DISK
a
SYSTEM
b

Pre-execution-time Tables or Arrays


a
DISK
b
SYSTEM
c

Execution-time Arrays

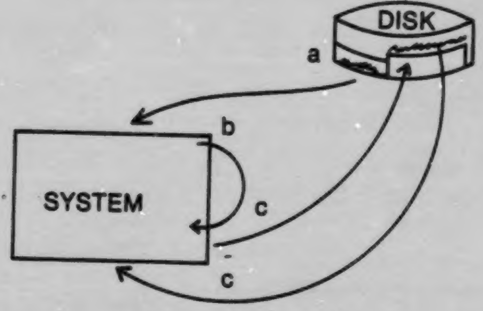

DISK
a
b
SYSTEM
c
c

Table and Array Data Operations

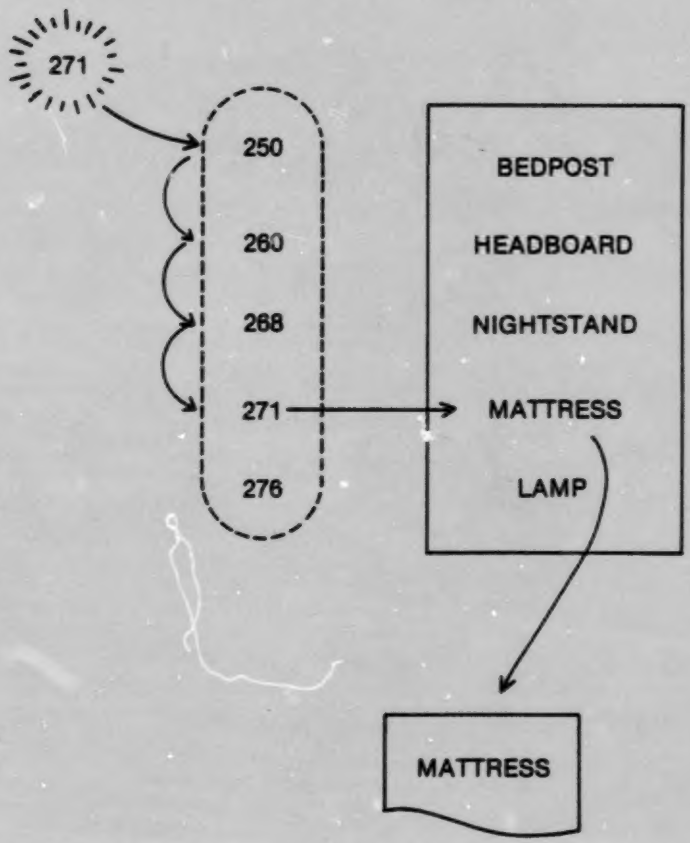
Lookup (LOKUP)

Use a known value

to search a table or array having similar values

and get a correspondingly located item from a second table or array

to be used for either calculations or output.

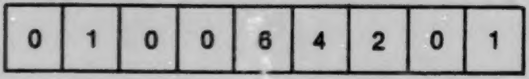

271
250
260
268
271
276
BEDPOST
HEADBOARD
NIGHTSTAND
MATTRESS
LAMP
MATTRESS

682

TM 4

Examples of XFOOT and MOVEA

XFOOT

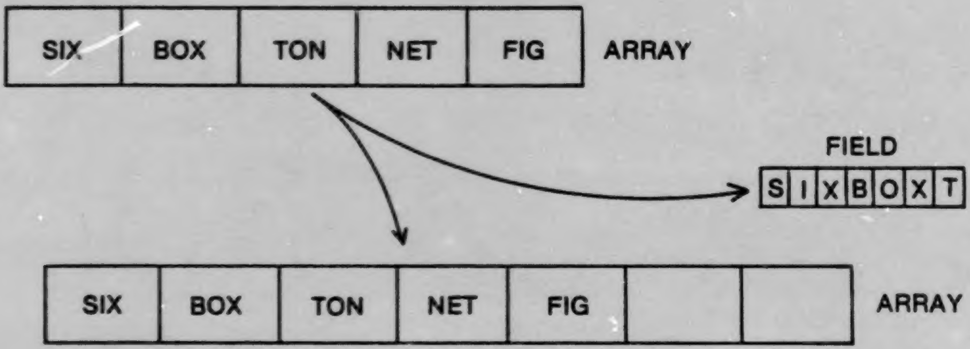


0	1	0	0	6	4	2	0	1

$0 + 1 + 0 + 0 + 6 + 4 + 2 + 0 + 1 = 14$

MOVEA

Array to Field or Another Array



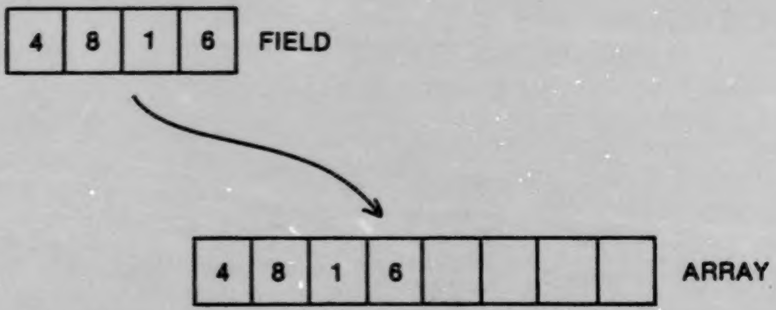
SIX	BOX	TON	NET	FIG	ARRAY

FIELD

S	I	X	B	O	X	T

SIX	BOX	TON	NET	FIG			ARRAY

Field to Array



4	8	1	6	FIELD

4	8	1	6					ARRAY

RPG II TOPICS: TABLES AND ARRAYS

UNIT XVI

ASSIGNMENT SHEET I: Sample Assignment Sheet Program

Note to students: This assignment sheet shows the process of coding a program that uses 2 tables. The coded solution and explanation are included so that you can study and understand the process for completing other assignments.

Directions: This sample assignment sheet program involves the printing of a list of items as ordered by customers. The list is to look like the sample report. One set of table data contains 50 item number entries, one for each item that is for sale. A second set of table data contains 50 corresponding cost entries. Here is a partial list of table data from both of them.

Table of Items	Table of Costs
38362	2.50
27865	7.80
79638	9.10
32176	1.15

Input records contain the item number and the quantity sold. In order to print the report, it is necessary to look up the unit price for the item and then multiply this price by the quantity to determine the amount of the sale.

REPORT

Sales

Item Number	Quantity	Unit Price	Sale Price
27865	100	7.80	780.00
32176	50	1.15	57.50
38362	740	2.50	1850.00

NOTE: When table data are used to solve a problem, we need to describe the table or tables being used and describe any calculations and/or output associated with this data. Tables are described on an Extension specification sheet. A table look-up operation (LOKUP) is used on the calculation sheet to search one table in order to find a corresponding table value in the other. After a desired value is found, it may be used for additional calculations and/or output. The coded solution and explanation are as follows:

Coded Solution:

Explanation of Assignment Sheet 1 Solution

1. File Description - two files are described. Orders are keyed and a report is printed.

2. Extension - two tables (TABITM and TABCST) are described. Table names always start with TAB in RPG II. Included in the description of tables is the "Length of Entry" and the "Number of Entries Per Table."

   For numeric table entries, include the number of "Decimal Positions."

   NOTE: We will discuss other specifications used on the Extension Specifications sheet later on in this chapter.

3. Input - the keyed record and its fields (ITEM# and QTY) are described.

4. Calculations - use an input field (ITEM#) as Factor 1, specify the table look-up operation (LOKUP), the table to be searched (TABITM) as Factor 2, the table containing corresponding unit prices (TABCST) as the Result Field, and assign an unused indicator for the desired search condition.

   In our example we are searching for an "Equal" condition, so indicator 40 was assigned in positions 58, 59. (See the headings for lookup above positions 54-59.)

   The second calculation (multiply unit price by number of items) is to be done when a desired table entry is found, so it is controlled by indicator 40. The cost value in the corresponding table (TABCST) is multiplied by the quantity (QTY) to determine the Sales price (SALE) of the item we keyed as input (ITEM#).

5. Output - four fields are to be printed when the unit price is found for the keyed item.

   The indicators used to control the printing of the detail line are 0 and 40 to be sure that a line is printed when information about the keyed record (ITEM#) is found in the table (TABCST).

RPG II TOPICS: TABLES AND ARRAYS

UNIT XVI

ASSIGNMENT SHEET 2: Coding Extension Sheets I

Directions: Answer the Review Questions in the space provided and complete the exercises as listed.

1. Determine which of the following table names are "acceptable" or "not acceptable" by placing a check in the appropriate column. If not acceptable, explain why.

	Acceptable	Not Acceptable	Reason not acceptable
TAB123			
TA55			
TAB			
TABX			
ABCDEF			
TAB/B			
TAB29			

2. Use an Extension sheet to define the following tables.

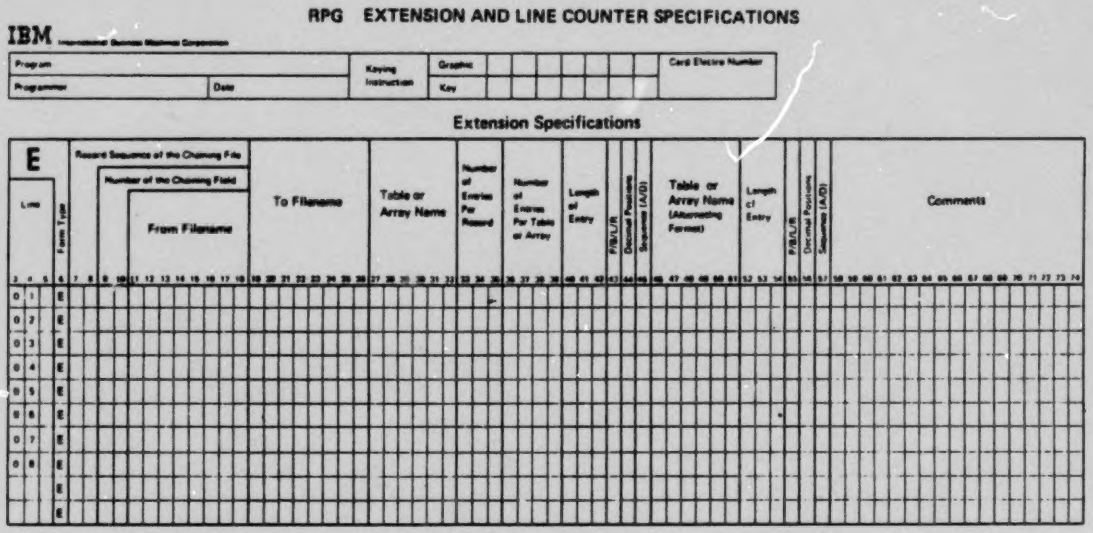
Name	**Type**	**Length of Entry**	**# of Entries Per Table of Array**	**# of Entries Per Record**
TABXYZ	numeric	3	16	16
TAB2	alphameric	20	25	4
TABC	numeric	7 with 2 decimals	100	14
TAB19	numeric	15 with 9 decimals	10	6

Remember:

- * Specify a table name starting in position 27.
- * Specify the "Length of Entry" in positions 40-42. If the entry is a numeric field, include the number of decimal positions in position 44.
- * Specify the "Number of Entries per Table or Array" in positions 36-39.
- * Specify the "Number of Entries per Record" in positions 33-35. For example, if each record containing table data was 100 positions long and each entry in that table was 5 positions long, there would be 20 "entries per record."

ASSIGNMENT SHEET 3: Coding Extension Sheets II

Directions: On the Extension specifications sheet that is shown below, program a table that contains the names of the months of the year. Assume that all table entries will fit into one record. Make up your own table name.



RPG EXTENSION AND LINE COUNTER SPECIFICATIONS

IBM International Business Machines Corporation

Program		Keying Instruction	Graphic		Card Electro Number
Programmer	Date		Key		

Extension Specifications

Line (3–5)	Form Type (6)	Record Sequence of the Chaining File (7–8)	Number of the Chaining Field (9–10)	From Filename (11–18)	To Filename (19–26)	Table or Array Name (27–32)	Number of Entries Per Record (33–35)	Number of Entries Per Table or Array (36–39)	Length of Entry (40–42)	P/B/L/R (43)	Decimal Positions (44)	Sequence (A/D) (45)	Table or Array Name (Alternating Format) (46–51)	Length of Entry (52–54)	P/B/L/R (55)	Decimal Positions (56)	Sequence (A/D) (57)	Comments (58–74)
01	E																	
02	E																	
03	E																	
04	E																	
05	E																	
06	E																	
07	E																	
08	E																	
	E																	
	E																	

687

BEST COPY AVAILABLE

RPG II TOPICS: TABLES AND ARRAYS

UNIT XVI

ASSIGNMENT SHEET 4: Describing a Pre-execution-time Table

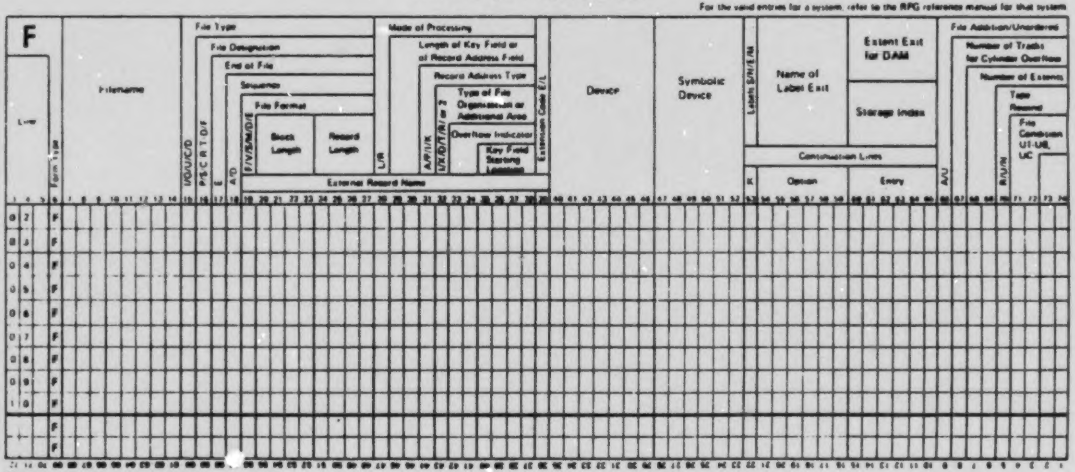
Directions: Describe a pre-execution-time table file having the following characteristics on a File Description sheet and an Extension sheet provided below.

1. The input table file has a record length of 54 characters and is stored on disk.
2. The table entries are numeric and six positions long with no decimals. There are 9 entries in each record and a total of 100 entries in the table.

Make up your filename and table name.

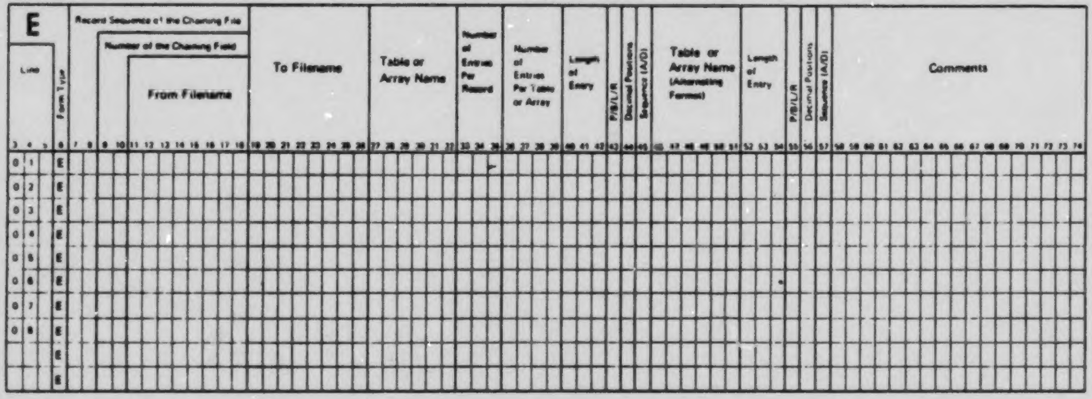
File Description Specifications

For the valid entries for a system, refer to the RPG reference manual for that system.


F
Line
Form Type
Filename
File Type
File Designation
End of File
Sequence
File Format
Block Length
Record Length
Mode of Processing
Length of Key Field or of Record Address Field
Record Address Type
Type of File Organization or Additional Area
Overflow Indicator
Key Field Starting Location
Extension Code E/L
External Record Name
Device
Symbolic Device
Labels S/N/E/M
Name of Label Exit
Extent Exit for DAM
Storage Index
Continuation Lines
Option
Entry
A/U
File Addition/Unordered
Number of Tracks for Cylinder Overflow
Number of Extents
Tape Rewind
File Condition U1-U8, UC

*Number of sheets per pad may vary slightly.

Extension Specifications


E
Line
Form Type
Record Sequence of the Chaining File
Number of the Chaining Field
From Filename
To Filename
Table or Array Name
Number of Entries Per Record
Number of Entries Per Table or Array
Length of Entry
P/B/L/R
Decimal Positions
Sequence (A/D)
Table or Array Name (Alternating Format)
Length of Entry
P/B/L/R
Decimal Positions
Sequence (A/D)
Comments

688

BEST COPY AVAILABLE

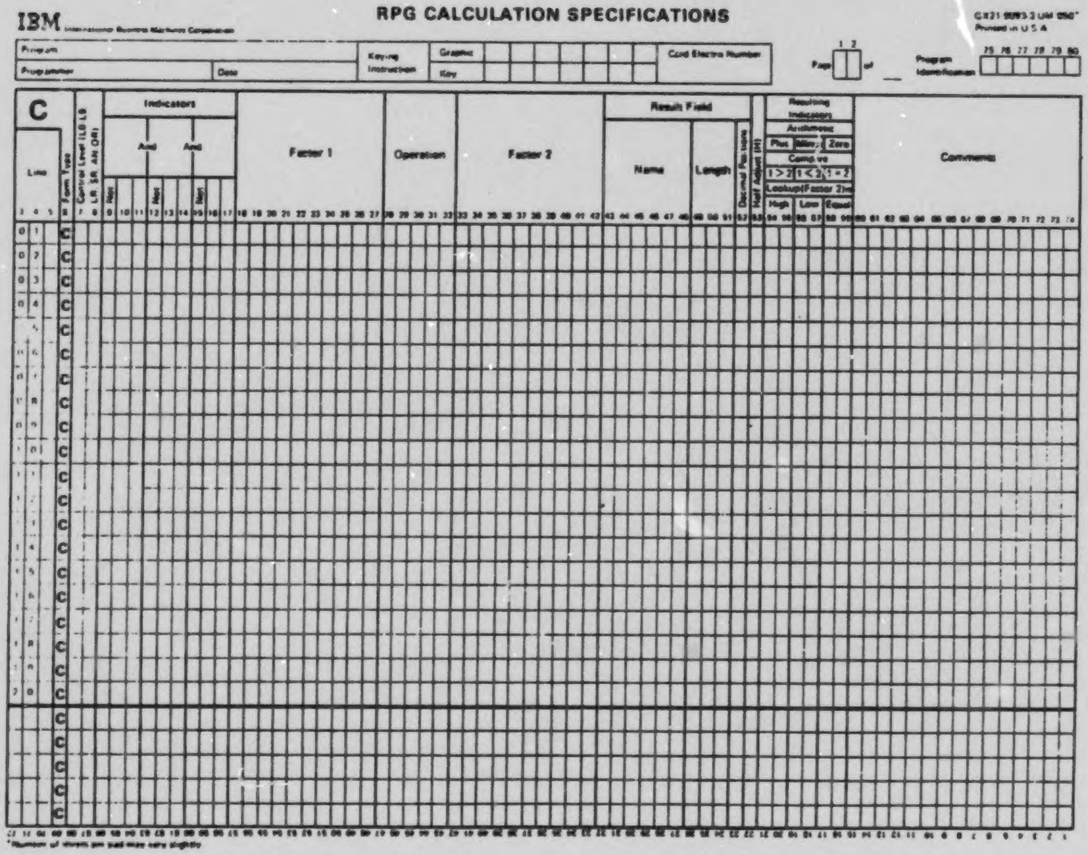
RPG II TOPICS: TABLES AND ARRAYS

UNIT XVI

ASSIGNMENT SHEET 5: Searching a Table

Directions: Write a program on the Specifications sheet provided below that searches a table named TABRAT for a value equal to the value in the search field named RATE. When an equal is found, also locate a corresponding value in a table named TABAMT.

(NOTE: Assign an indicator in positions 58-59 to test the look-up operation for an equal condition.)


IBM
RPG CALCULATION SPECIFICATIONS
Program
Programmer
Date
Keying Instruction
Graphic
Key
Card Electro Number
Page
of
Program Identification
75 76 77 78 79 80
C
Line
Form Type
Control Level (L0-L9, LR, SR, AN/OR)
Indicators
And
And
Not
Factor 1
Operation
Factor 2
Result Field
Name
Length
Decimal Positions
Half Adjust (H)
Resulting Indicators
Arithmetic
Plus
Minus
Zero
Compare
1 > 2
1 < 2
1 = 2
Lookup(Factor 2)is
High
Low
Equal
Comments

RPG II TOPICS: TABLES AND ARRAYS

UNIT XVI

ASSIGNMENT SHEET 6: Finding a Value in a Table

Directions: In this assignment, you are to find a value in a table that is the first entry which is higher than the search field value.

Obviously the entries in the table need to be in some sort of sequence or we could not find a "higher" entry. Also, the entries do not include every possible value. For example, let's say we have a numeric table with these values.

TABCUB

001
008
027
064
125
216

NOTE: When the look-up is completed and we are searching for a "high" condition, the program compares the value in the search field (assume it is 025) against the first table value.

1. Is factor 2 (the first table value) high?

2. Is the second table value high?

3. If the second table value isn't high, what happens?

4. Is the table value 27 high compared to 025? _____ What happens now?

5. What will happen the next time if the new search field value is 220?

6. Will an indicator come on if the new search field value is 216?

RPG II TOPICS: TABLES AND ARRAYS

UNIT XVI

7. Examine the following coding.

RPG CALCULATION SPECIFICATIONS

Line	Form Type	Control Level (L0-L9, LR, SR, AN/OR)	Indicators	And	And	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Half Adjust (H)	Resulting Indicators High (1>2)	Low (1<2)	Equal (1=2)
01	C					SEARCH	COMP	216					17		17
02	C		17				GOTO	PASS							
03	C					SEARCH	LOKUP	TABCUB					48		
04	C														
05	C														
06	C														
07	C														
08	C					PASS	TAG								
09	C														

Now circle the letter preceding the correct answer from the following choices:

a. Indicator 17 will never be turned on.

b. If indicator 48 is turned on, SEARCH is less than 216.

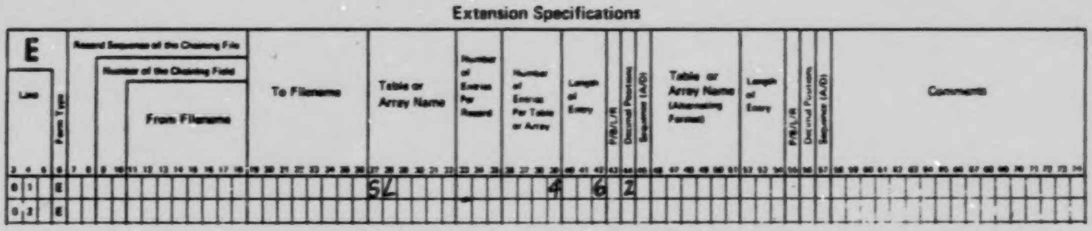
c. If indicator 17 is turned on, indicator 48 will also be turned on.

RPG II TOPICS: TABLES AND ARRAYS

UNIT XVI

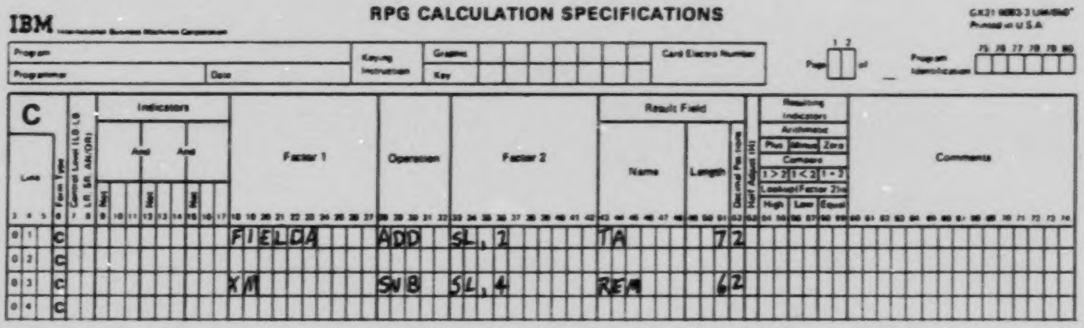
ASSIGNMENT SHEET 7: Array Problem and Questions

Directions: Look at the following coding sheets and answer the questions below.



Extension Specifications

Line	Form Type	From Filename	To Filename	Table or Array Name	Number of Entries Per Record	Number of Entries Per Table or Array	Length of Entry	P/B/L/R	Decimal Positions	Sequence (A/D)	Comments
01	E			SL	–	4	6		2		
02	E										



IBM RPG CALCULATION SPECIFICATIONS

Line	Form Type	Control Level	Indicators	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Half Adjust (H)	Resulting Indicators	Comments
01	C			FIELDA	ADD	SL,1	TA	7	2			
02	C											
03	C			XM	SUB	SL,4	REM	6	2			
04	C											

1. What is the name of the array?

2. How big is each entry?

3. How many entries are there in the array?

4. Refer to examples on the following page. Place answers in appropriate spaces.

   What does each of these steps mean?

   How did you interpret each example?

Remember, the use of an index value and an array name references a particular entry in that array and treats it as though it were a field of data.

BEST COPY AVAILABLE

BEST COPY AVAILABLE

RPG II TOPICS: TABLES AND ARRAYS

UNIT XVI

ASSIGNMENT SHEET 7 (cont'd):

a.

b.

c.

d.

C

Line	Form Type	Control Level	Indicators	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Half Adjust (H)	Resulting Indicators High	Low	Equal
01 A	C			SL,1	MULT	6	X						
02 B	C			SL,3	LOKUP	TABS	TAB6						29
03 C	C				MOVEA	AR	SL,2						
04	C												

O

Line	Form Type	Filename or Record Name	Output Indicators	Field Name or EXCPT Name	Edit Codes	End Position in Output Record	Constant or Edit Word
01 D	O			SL,2	1	45	
02	O						

5. Which 5 combinations of entries for Resulting Indicators are acceptable with the LOKUP operation on the Calculation sheet?

6. How do I reference a particular entry of an array?

7. An index for an array may be either a(n) ______________________________ or a(n) ______________________.

633

RPG II TOPICS: TABLES AND ARRAYS

UNIT XVI

ANSWERS TO ASSIGNMENT SHEETS

Assignment Sheet 1:

See coded solution and explanation on AS 1.

Assignment Sheet 2:

1. Acceptable table names are: TAB123, TABX, and TAB. Why are the others not acceptable?

   TAB/B has a special character. TA55 does not follow the rule, "Start with TAB." ABCDEF is wrong for the same reason. TAB 29 has a blank which is not a letter or a number.

2. Your entries should look like this.

   (NOTE: Pass out extension sheets.)

Assignment Sheet 3:

If your coding is correct, you have TAB in positions 27-29. Positions 30-32 may be blank or have any letters or numbers (without a blank between them). Positions 34-35 should contain a 12 and positions 38-39 should contain 12. The length of a table entry is always large enough to hold the longest word or phrase or the largest number to be stored in the table. Since the longest name is SEPTEMBER, your entry for positions 40-42 should be 9 in position 42.

Assignment Sheet 4:

Check your coding against these entries. If you do not know why yours does not agree, reread this section about "Pre-execution-time" tables.

Assignment Sheet 5:

Here are my solutions. Your indicator entry may be different, but the other entries must be the same as mine.

Assignment Sheet 6:

1. No, because the first entry in the table is only 001.
2. No.
3. It looks at the next table value.

4. Yes, 027 is high as compared to 025. The table look-up operation is complete and the indicator assigned in position 54-55 for a high condition turns on.

5. In this case, the indicator is not turned on because the table value is never higher than the search value.

6. No, because the table value is not higher.

7. The correct choice is number 2. When examining table data and possible search field values, consider the use of a compairson to bypass the steps that follow. You might also wish to halt processing under certain conditions.

Assignment Sheet 7:

1. The array is named SL.

2. 6 positions long with 2 decimals.

3. Contains 4 numeric entries

   Notice that the first calculation step describes the additions of a field (FIELDA) to "something" (SL,1) to get a result (TA). Also, the second calculation step describes the subtraction of "something" (SL,4) from a field (XM) to get an answer called REM. In each case, the "something" included the name of the array followed by a comma and a constant number. Here is what each step means.

4. SL,1 refers to the use of data in the first entry in array SL for the addition step. What do you think SL,4 means?

   It means, refer to the use of data in the fourth entry in array SL for the subtraction step.

   a. Multiply the value in the first entry (SL,1) of array SL by 6 to get X.

   b. Search the table named TAB5 for a value equal to the value in the third entry (SL,3) of array SL. If an equal is found, locate the corresponding value in TAB6.

   c. Move the characters in array AR in the left most positions of the second entry (SL,2) of array SL. Stop the movement of data when either the second entry of SL is filled or when all characters in array AR have been moved.

   d. When producing output, include the first entry (SL,1) of array SL as a field using the edit code 1 to include commas. The right most character shall be in output position 48.

5. High, High and Equal, Low, Low and Equal, and Equal

6. I use an index value to reference a particular array entry.

7. An index for an array may be either a field or a literal.

RPG II TOPICS: TABLES AND ARRAYS

UNIT XVI

EXAMINATION

1.1. Define a table.

1.2. What are the five rules used to describe tables?

a.

b.

c.

d.

e.

2.1. When table data is included as a part of your RPG II source program, it is referred to as a ____ .

2.2. What is the extension sheet used for?

3. When using pre-execution time tables, what entries must be made on the file description sheet.

a.

b.

4. What does the operation LOKUP accomplish?

5.1. Search a table named TABTAX for a value equal to the value 175. When an equal is found, also locate a corresponding value in TABSTX.

C Line 3 4 5	Form Type 6	Control Level (L0-L9, LR, SR, AN/OR) 7 8	Indicators Not And Not And Not 9–17	Factor 1 18–27	Operation 28–32	Factor 2 33–42	Result Field Name 43–48	Result Field Length 49–51	Decimal Positions 52	Half Adjust (H) 53	Resulting Indicators Arithmetic Plus Compare 1 > 2 Lookup(Factor 2)is High 54 55	Resulting Indicators Arithmetic Minus Compare 1 < 2 Lookup(Factor 2)is Low 56 57	Resulting Indicators Arithmetic Zero Compare 1 = 2 Lookup(Factor 2)is Equal 58 59
0 1	C												
0 2	C												

5.2. Below are terms dealing with similarities and differences in Tables and Arrays. Place the appropriate letter in the blank across from the proper term, using (T) for Table, (A) for Array, and (B) for Both.

____ a. XFOOT

____ b. LOKUP

____ c. Extension sheet

____ d. Name begins with TAB

____ e. Indexing

____ f. MOVEA

____ g. Execution-time

____ h. Compile-time

____ i. Pre-execution-time

5.3. List coding sheets and describe necessary operations to print out the contents of a table or array at the end of a job.

a.

b.

c.

RPG II TOPICS: TABLES AND ARRAYS

UNIT XVI

ANSWERS TO THE EXAMINATION

1.1. A table is an arrangement of data items having like characteristics.

1.2. a. Every table entry must be the same length.

b. Table entries must either be numeric or alphameric.

c. Every entry in a numeric table must have same number of decimal positions.

d. Every table must start with TAB as its name.

e. Each table is described on the extension specification sheets.

2.1. Compile time table

2.2. To describe every table used in a program.

3. a. T in position 16

b. An E in position 39

4. It initiates a search of table or array entries to find an acceptable "match" based upon the condition(s) tested by Resulting Indicators. When a search is successful, the particular entry of the table or array is stored for use in further calculations or output.

5.1.

C Line 3-5	Form Type 6	Control Level (L0-L9, LR, SR, AN/OR) 7-8	Indicators (Not, And, Not, And, Not) 9-17	Factor 1 18-27	Operation 28-32	Factor 2 33-42	Result Field Name 43-48	Result Field Length 49-51	Decimal Positions 52	Half Adjust (H) 53	Resulting Indicators (Arithmetic: Plus, Minus, Zero; Compare: 1>2, 1<2, 1=2; Lookup(Factor 2) is: High) 54-55	Low 56-57	Equal 58-59
01	C			175	LOKUP	TABTAX	TABSTX						20
02	C												

(NOTE: Your indicator entry may be different, but the other entries must be the same.

5.2. a. A
b. B
c. B
d. T
e. A
f. A
g. A
h. B
i. B

5.3. a. Use a File Description sheet to describe an Output file to hold the table or array.

b. Use the Extension Sheet to include the filename in positions 19-26.

c. Describe total-time output records with LR indictor for last record. Include the name of the table or array in position 32-37.

DATA ENTRY AND EDIT PROGRAMS

UNIT XVII

OBJECTIVES

After completion of this unit, the student should understand batch data entry and the programming required to edit input data.

SPECIFIC OBJECTIVES

After studying this unit, the student should be able to:

1. Explain the function of data entry.
2. Describe a typical data processing routine that is performed daily in a small business.

3.1 Recall how edit listing is used.

3.2 List errors detected by edit listing.

4. Complete Data Entry and edit Program 1.
5. Complete Data Entry and edit Program 2.

700

DATA ENTRY AND EDIT PROGRAMS

UNIT XVII

SUGGESTED ACTIVITIES

I. Instructor

  A. Provide students with objectives and discuss.

  B. Provide students with information and assignment sheets and discuss.

  C. Discuss unit and specific objectives.

  D. Discuss information sheets.

  E. Provide handouts and assignment sheets.

  F. Discuss assignment sheets.

II. Student

  A. Read objectives.

  B. Study information sheets and handouts.

  C. Complete assignment sheets.

INSTRUCTIONAL MATERIALS

I. Unit content

  A. Objectives

  B. Information sheets

  C. Transparency master

    1. TM 1: Data Processing Routine

  D. Assignment sheets

    1. AS 1: Program 1 Description
    2. AS 2: Program 2 Description

  E. Answer to assignment sheets

  F. Examination

  G. Answer to examination

II. Reference

  A. IBM SYSTEM 34/36/38 RPG II Reference Manual. Information Development. Rochester, MN: IBM Corp., 1983.

DATA ENTRY AND EDIT PROGRAMS

UNIT XVII

NOTES TO THE INSTRUCTOR

I. Before presenting Assignment Sheet 1 to the students, the instructor should:

  A. Study order transaction file layout for Ship To Record, Item Record, and Customer Header Record types.

  B. Create transaction XACT file on disk or diskette for student use when they run the programs on the system. Records must be keyed in exactly as shown for proper report format.

  C. When viewing required output, look at entry data sheet for Record 1, Record 2, Record 3, etc. This will help identify which records are being read and how they are printed on the report.

  D. Read Program 1 Description carefully to the students point by point. Remind them that what looks overwhelming at first can be broken down and segmented into smaller and easier tasks.

II. Before presenting Assignment 2 to the students, the instructor should:

  A. Realize transaction (XACT) file will remain the same.

  B. Have students create proper master file and customer file on system. Tell students to key records in exactly as shown. (These files are indexed and will be accessed by the program to verify necessary information to see if data was keyed properly.)

  C. Read Program 2 Description carefully and go over required output. Then instruct students to start the project.

DATA ENTRY AND EDIT PROGRAMS

UNIT XVII

Information Sheet

I. Function of Data Processing in the machine readable form

   A. Captures information in a medium which can be used repeatedly

   B. Encodes information in a standard format which can be processed by a machine

II. Typical Data Processing Routine (TM 1)

   A. Event - information is created by an event.

   B. Transaction - the event is recorded on a source document so that information can be passed from one process to another.

   C. Batch

      1. All transactions are batched (grouped) into a convenient size for further processing.

         a. Control tape

            1) Each batch is totaled to assure accuracy.

            2) Hash totals are used as controls for all critical numeric fields.

            3) Simple totals or totals are used for numeric fields which are normally totaled.

      2. Transactions may be transferred to a special keypunching form (card, diskette, or key to disk from a terminal) as part of the batching operation. They are commonly found when data entry loads are high and user departments are not responsible for data entry.

   D. Keypunch - converts source information into machine readable form (card, disk, or diskette)

   E. Verify - rekeys critical information to assure accuracy

   F. Process - loads machine readable information into a machine for processing

   G. Edit Listing - lists all information with the last page of the report containing machine generated totals and hash totals used to check against original control tape to assure accuracy

III. Edit Listing

A. Use - to detect errors in data entry prior to invoicing and updating master records

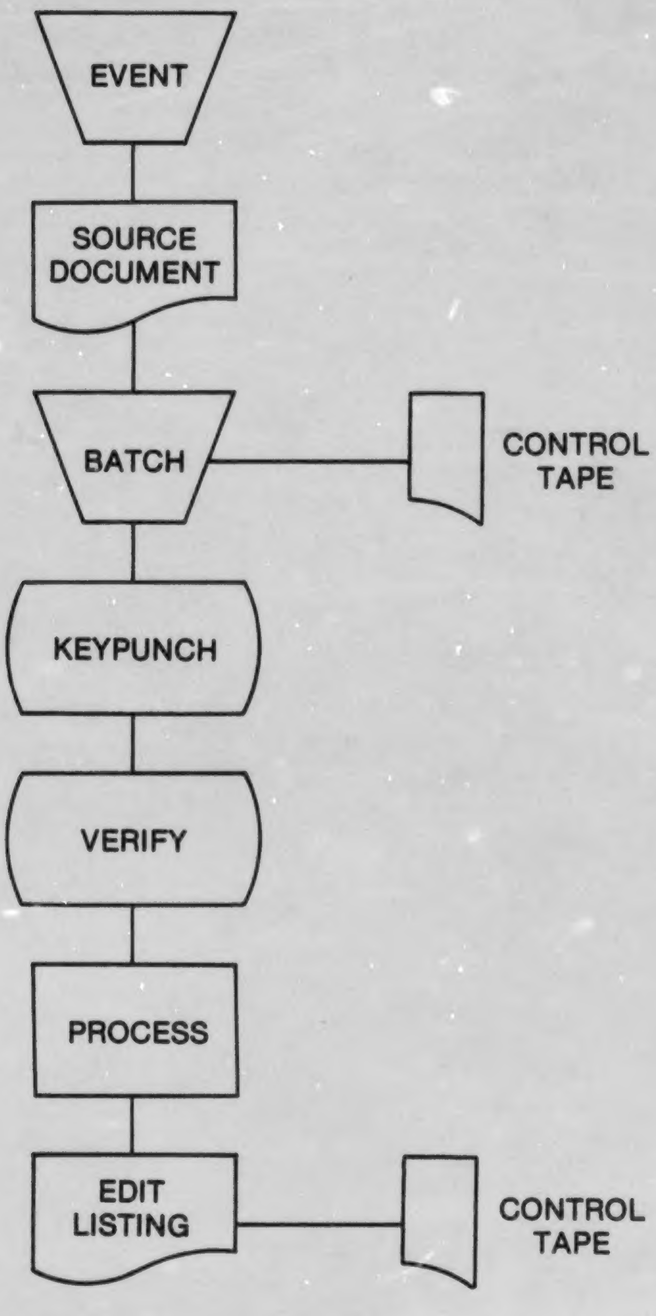
B. Content

1. Is the customer number valid?
2. Is the item number valid?
3. Is the item number missing?
4. Is the valid price overridden?
5. Does the invoice profit percent meet company objectives?

C. Advantages - improves accuracy of invoices

TM 1

Data Processing Routine


EVENT
SOURCE DOCUMENT
BATCH
CONTROL TAPE
KEYPUNCH
VERIFY
PROCESS
EDIT LISTING
CONTROL TAPE

DATA ENTRY AND EDIT PROGRAMS

UNIT XVII

ASSIGNMENT SHEET I: PROGRAM I DESCRIPTION

Directions: Read and code problem on appropriate coding sheets.

Input:

- XACT file (See Part #1)

Output:

- Report 'Edit Report I' (See Part #3)

1. The report should look exactly like the attached example. Horizontal (spaces between fields) positions may vary slightly but vertical spacing (spaces between lines) must be exact.

2. Messages must be printed if the following fields are blank:

   Header Record (CU) (See Part #2 A)
      Order number
   Ship-To (SH) (See Part #2 B)
      -Order number
      -Zip code
   Item (IT) (See Part #2 C)
      Item number
      Quantity ordered

3. Messages must also print if:

   - any quantity ordered field exceeds 1000
   - no item records exist for an order
   - the header record for an order is missing

4. The entire record must be printed for records containing invalid record codes, with the message 'INVALID RECORD CODE' printed on the next line.

5. Order and final totals will be printed for the following fields:

   - Item number
   - Quantity ordered
   - Price
   - Quantity shipped

6. The bill-to and ship-to lines must be printed only if that data record is present.

PART #1: TRANSACTION FILE DATA

ENTRY DATA FILE NAME: XACT

Record 1

Record Code	CU
Order Number	000001
Customer Number	000981
Customer Name	
Add. Line 1	
Add. Line 2	
Zip	
P.O. Number	PP88117
Order Date	041477

Record 2

Record Code	SH
Order Number	000001
Ship Name	Lamkin & Associates
Ship Add. Line 1	Canyon Estates
Ship Add. Line 2	Carrollton, MO
Ship Zip	19768
Ship Inst.	C.O.D.

Record 3

Record Code	IT
Order Number	000001
Item Number	00000166
Quantity Ordered	00002
Price	00000
Quantity Shipped	00000
Description	

Record 4

Record Code	IT
Order Number	000001
Item Number	00000229
Quantity Ordered	00016
Price	00000
Quantity Shipped	00000
Description	

Record 5

Record Code	IT
Order Number	000001
Item Number	00000000
Quantity Ordered	00004
Price	00300
Quantity Shipped	00004
Description	Metal Can 1 gallon

Record 6

Record Code	IT
Order Number	000001
Item Number	00000415
Quantity Ordered	00026
Price	00000
Quantity Shipped	00000
Description	

Record 7

Record Code	IT
Order Number	000001
Item Number	0000041
Quantity Ordered	00100
Price	
Quantity Shipped	
Description	

Record 8

Record Code	IT
Order Number	000001
Item Number	00000422
Quantity Ordered	00002
Price	
Quantity Shipped	
Description	

*Record 9

Record Code	CU
Order Number	000002
Customer Number	000990
Customer Name	Minard Steel Fabrication
Add. Line 1	2509 Northern Hill Ct
Add. Line 2	Atco NJ
Zip	29876
P.O. Number	PP92KL
Order Date	041477

Record 10

Record Code	IT
Order Number	000002
Item Number	00000508
Quantity Ordered	00020
Price	00000
Quantity Shipped	00000
Description	

Record 11

Record Code	IT
Order Number	000002
Item Number	00000196
Quantity Ordered	00006
Price	
Quantity Shipped	
Description	

Record 12

Record Code	IT
Order Number	000002
Item Number	00000197
Quantity Ordered	00008
Price	
Quantity Shipped	
Description	

Record 13

Record Code	IT
Order Number	000002
Item Number	00000260
Quantity Ordered	00006
Price	
Quantity Shipped	
Description	

Record 14

Record Code	CU
Order Number	000003
Customer Number	000971
Customer Name	
Add. Line 1	
Add. Line 2	
Zip	00000
P.O. Number	PP654BTY
Order Date	041477

*Record 15

Record Code	SH
Order Number	000003
Ship Name	Schmidt Steel Fabrication
Ship Add. Line 1	2509 Northern Hill Ct
Ship Add. Line 2	Atco NJ
Ship Zip	29879
Ship Inst.	C.O.D.

Record 16

Record Code	IT
Order Number	000003
Item Number	00000197
Quantity Ordered	
Price	145.95
Quantity Shipped	
Description	1300CFM Fan (Rebuilt)

*Customer name and address Line 1 fields in the records are the same as any other records. However, due to amount of data in these fields, it caused our typesetting to be realigned slightly on page.

Record 17

Record Code	IT
Order Number	000003
Item Number	00000000
Quantity Ordered	12000
Price	00000
Quantity Shipped	00000
Description	

Record 18

Record Code	IT
Order Number	000003
Item Number	00000415
Quantity Ordered	00017
Price	00000
Quantity Shipped	00000
Description	

Record 19

Record Code	CU
Order Number	000000
Customer Number	000981
Customer Name	
Add. Line 1	
Add. Line 2	
Zip	00000
P.O. Number	PP88117
Order Date	041477

Record 20

Record Code	CU
Order Number	000004
Customer Number	000941
Customer Name	
Add. Line 1	
Add. Line 2	
Zip	00000
P.O. Number	
Order Date	041477

*Record 21

Record Code	SH
Order Number	000004
Ship Name	O'Brien Mfg. Co.
Ship Add. Line 1	5855 Monroe Street
Ship Add. Line 2	Toledo OH
Ship Zip	00000
Ship Inst.	

Record 22

Record Code	IT
Order Number	000004
Item Number	00000508
Quantity Ordered	00005
Price	
Quantity Shipped	
Description	

Record 23

Record Code	IT
Order Number	000004
Item Number	00000196
Quantity Ordered	00000
Price	00000
Quantity Shipped	00000
Description	

Record 24

Record Code	IT
Order Number	000004
Item Number	00000197
Quantity Ordered	00006
Price	
Quantity Shipped	
Description	

*Customer name and add. Line 1 fields in these records are the same as any other records. However, due to amount of data in these fields, it caused our typesetting to be realigned slightly on page.

Record 25

Record Code	IT
Order Number	000004
Item Number	00000260
Quantity Ordered	00006
Price	
Quantity Shipped	
Description	

Record 26

Record Code	IT
Order Number	000004
Item Number	00000217
Quantity Ordered	00002
Price	
Quantity Shipped	
Description	

Record 27

Record Code	DU
Order Number	000005
Customer Number	000901
Customer Name	
Add. Line 1	
Zip	00000
P.O. Number	AD44496
Order Date	041477

Record 28

Record Code	SH
Order Number	000005
Ship Name	Schnare & Moller Inc.
Ship Add. Line 1	Two Penn Plaza
Ship Add. Line 2	New York, NY
Ship Zip	22508
Ship Inst.	ASAP

Record 29

Record Code	IT
Order Number	000005
Item Number	00000415
Quantity Ordered	00020
Price	
Quantity Shipped	
Description	

Record 30

Record Code	IT
Order Number	000005
Item Number	00000422
Quantity Ordered	00001
Price	
Quantity Shipped	
Description	

Record 31

Record Code	CU
Order Number	
Customer Number	
Address Line 1	
Address Line 2	
Zip	
P.O. Number	
Order Date	

PART #2A: ORDER TRANSACTION FILE - RECORD CODE (CU)

File Name - XACT
Organization - Sequential
Record Length - 128
Number of record types - 3
Header record one per order

Description	From	To	Length	Type
Record Code (CU)	1	2	2	A
Order Number	3	8	6/0	N
Customer Number	9	14	6/0	N
Customer Name	15	39	25	A
Customer Address Line 1	40	64	25	A
Customer Address Line 2	65	89	25	A
Customer Address Line 3	90	109	20	A
Zip Code	110	114	5/0	N
Purchase Order Number	115	122	8	A
Order Date	123	128	6	N

PART #2B: ORDER TRANSACTION FILE - RECORD CODE (SH)

File Name - XACT
Organization - Sequential
Record Length - 128
Number of record types - 3
Ship-to address record one per order

Description	From	To	Length	Type
Record Code (SH)	1	2	2	A
Order Number	3	8	6/0	N
Ship Name	9	33	25	A
Ship Address Line #1	34	58	25	A
Ship Address Line #2	59	83	25	A
Shipping Instructions	84	103	20	A
Zip Code	104	108	5/0	N

PART #2C: ORDER TRANSACTION FILE - RECORD CODE (IT)

File Name - XACT
Organization - Sequential
Record Length - 128
Number of record types - 3
Ship-to address record one per order

Description	From	To	Length	Type
Record Code (IT)	1	2	2	A
Order Number	3	8	6/0	N
Item Number	9	16	8/0	N
Quantity Ordered	32	36	5/0	N
Price	37	41	5/2	N
Quantity shipped	56	60	5/0	N
Description	61	87	27	A

713

ASSIGNMENT SHEET 2: PROGRAM 2 DESCRIPTION

Directions: Read and code problem on appropriate coding sheets.

Input:

- XACT file	
- CMAST file	(See Part #1)
- ITEMMST file	(See Part #2)

Output:

- Report 'Edit Report 2' (See Part #3)

1. The report format is identical to that of Program 1.
2. Program 2 will verify the item and customer numbers by chaining to the appropriate file.
3. Price, description, and customer name will be printed from the master files if not in the transaction file.
4. The order value will be calculated by multiplying the quantity ordered by the price in the master file unless an override price has not been keyed.
5. The order value plus the customer balance will be used to determine if the credit limit amount has been exceeded and by how much.
6. The control totals for the price field will be dropped.

PART #1: CUSTOMER MASTER FILE - Record Code (MA)

Record Length - 256 characters
File Name - CMAST
Organization - Indexed
Key Start - 3
Key Length - 6
Number of record types - 1

Description	From	To	Length	Type
Record Code (MA)	1	2	2	A
Customer Number	3	8	6/0	N
Customer Name	9	33	25	A
Customer Address Line #1	34	58	25	A
Customer Address Line #2	59	83	25	A
Customer Address Line #3	84	103	20	A
Zip Code	104	108	5/0	N
Credit Limit Amount	136	140	5/0	N
Balance	171	177	7/2	N
Ship-To Code	178	178	1	A

PART #1 Con't.: CUSTOMER MASTER FILE - Record Code (MA)

CUSTOMER MASTER FILENAME: CMAST

Customer # 000941 (don't key in decimal point)

Record Code	MA
Name	Sinclair Dry Goods Co.
Address Line 1	8712 W. Dodge Road
Address Line 2	Omaha NE
Zip Code	29540
Credit Limit	03000
Balance	00400.00

Customer #000971

Record Code	MA
Name	Chapman Supply
Address Line 1	580 Walnut Street
Address Line 2	Cincinnati OH
Zip Code	13536
Credit Limit	03000
Balance	00700.00
Ship Code	

Customer #000981

Record Code	MA
Name	Satterfield Spirits Co.
Address Line 1	6128 E. 38th Street
Address Line 2	Tulsa OK
Zip Code	29512
Credit Limit	03000
Balance	00800.00
Ship Code	

PART #2: ITEM MASTER FILE - Record Code (IM)

File Name - ITEMMST
Organization - Indexed
Key Start - 4
Key Length - 8
Number of record types - 1
Record Length - 249 Characters

Description	From	To	Length	Type
Record Code (IM))	1	2	2	A
Delete Code (D)	3	3	1	A
Item Number	4	11	8/0	N
Item Description	32	58	27	A
List Price	77	80	4/2	N
Quantity Onhand	109	111	3/0	N

717

PART #2 Con't: ITEM MASTER FILE - Record Code (IM)

Item #166

Record Code	IM
Delete Code	
Description	Cup Dispenser
Price	2.50
Qty Onhand	200

Item #196

Record Code	IM
Delete Code	
Description	Milling Compound
Price	6.35 (No decimal pt. Keyed)
Qty Onhand	425

Item #197

Record Code	IM
Delete Code	
Description	3000 CFM Fan
Price	349.95
Qty Onhand	15

Item #229

Record Code	IM
Delete Code	
Description	Glass Jar 1 Qt.
Price	45.00
Qty Onhand	20

Item #230

Record Code	IM
Delete Code	
Description	Metal Can 1 Gal.
Price	12.75
Qty Onhand	50

Item #260

Record Code	IM
Delete Code	
Description	Shaft Assembly
Price	62.50
Qty Onhand	24

Item #415

Record Code	IM
Delete Code	
Description	Bottle of Wine
Price	2.85
Qty Onhand	250

Item #416

Record Code	IM
Delete Code	
Description	Wine Glasses
Price	23.50
Qty Onhand	400

Item #422

Record Code	IM
Delete Code	
Description	Sunken Bath
Price	200.50
Qty Onhand	2

Item #508

Record Code	IM
Delete Code	
Description	Line Metal Flange
Price	5.75
Qty Onhand	26

BEST COPY AVAILABLE

Part # 3: Edit Report I

```
DATE  5/20/83                                                                                  GF
              ORDER    CUSTOMER   CUSTOMER                                    ERROR MESSAGES
              NUMBER   NUMBER     NAME                       P.O. NUMBER

BILL TO            1   000981                                PP88117

SHIP TO            1              LANKIN & ASSOCIATES       SHIPPING INSTRUCTIONS
                                  CANYON ESTATES            C.O.D.
                                  CARROLLTON,MO.       19768

                    ITEM NUMBER     QTY     PRICE      QTY      DESCRIPTION
                                  ORDERED            SHIPPED
                        166           2       .00
                        229          16       .00
                                      4      3.00         4 METAL CAN 1 GALLON         ITEM NUMBER MISSING
                        415          26       .00
                        416         100       .00
                        422           2       .00

ORDER TOTALS           1648         150      3.00         4
*************************************************************************
BILL TO            2   000990     MINARD STEEL FABRICATION      PP92KL
                                  2509 NOTHERN HILL CT
                                  ATCO, N.J.

                    ITEM NUMBER     QTY     PRICE      QTY      DESCRIPTION
                                  ORDERED            SHIPPED
                        508          20       .00
                        196           6       .00
                        197           8       .00
                        260           6       .00

ORDER TOTALS           1161          40       .00         0
*************************************************************************
BILL TO            3   000971                                PP6548TY

SHIP TO            3              SCHMIDT STEEL FABRICATION SHIPPING INSTRUCTIONS
                                  2509 NOTHERN HILL CT      C.O.D.
                                  ATCO, N.J.           29879

                    ITEM NUMBER     QTY     PRICE      QTY      DESCRIPTION
                                  ORDERED            SHIPPED
                        197           5    145.95           1300CFM FAN  REBUILT
                                  12000       .00                                      ITEM NUMBER MISSING
                                                                                        QUANTITY ORDERED OVER1000
                        415          17       .00

ORDER TOTALS            612       12022    145.95         0
*************************************************************************
BILL TO                000981                                PP88117                   ORDER NUMBER BLANK

ORDER TOTALS              0           0       .00         0                            MISSING ITEM RECORDS
*************************************************************************
BILL TO            4   000941

SHIP TO            4              O'BRIEN MFG. CO.          SHIPPING INSTRUCTIONS
                                  5855 MONROE STREET
                                  TOLEDO, OH                                           ZIP CODE BLANK

                    ITEM NUMBER     QTY     PRICE      QTY      DESCRIPTION
                                  ORDERED            SHIPPED
                        508           5       .00



DATE  5/20/83
              ORDER    CUSTOMER   CUSTOMER                                    ERROR MESSAGES
              NUMBER   NUMBER     NAME                       P.O. NUMBER

                        196                   .00                                      QUANTITY ORDERED ZERO
                        197           6       .00
                        260           6       .00
                        217           2       .00
DU000005000901                                                                                00000AD44496 041477
                                                                                       INVALID RECORD CODE
ORDER TOTALS           1378          19       .00         0
*************************************************************************
           *************************NO BILL TO*************************                 MISSING HEADER RECORD

SHIP TO            5              SCHNAPE & MOLLER INC      SHIPPING INSTRUCTIONS
                                  TWO PENN PLAZA            ASAP
                                  NEW YORK, NY         22508

                    ITEM NUMBER     QTY     PRICE      QTY      DESCRIPTION
                                  ORDERED            SHIPPED
                        415          20       .00
                        422           1       .00

ORDER TOTALS            837          21       .00         0
*************************************************************************
BILL TO                000000                                                          ORDER NUMBER BLANK

ORDER TOTALS              0           0       .00         0                            MISSING ITEM RECORDS
*************************************************************************


                    ITEM NUMBER      QTY ORDERED     PRICE        QTY SHIPPED
FINAL TOTALS           5636            12252        148.95            4
```

BEST COPY AVAILABLE

Part # 3: Edit Report II

```
DATE  5/23/83
               ORDER   CUSTOMER   CUSTOMER                                                     ERROR MESSAGES
               NUMBER  NUMBER     NAME                         P.O. NUMBER

BILL TO            1  000981    SATTERFIELD SPIRITS CO.        PP88117

SHIP TO            1            LANKIN & ASSOCIATES            SHIPPING INSTRUCTIONS
                                CANYON ESTATES                 C.O.D.
                                CARROLLTON,MO.          19768

                      ITEM NUMBER    QTY      PRICE     QTY       DESCRIPTION
                                   ORDERED            SHIPPED
                           166         2                     CUP DISPENSER
                           229        16                     GLASS JAR 1 QT.
                                       4                   4 METAL CAN 1 GALLON             ITEM NOT ON MASTER FILE
                           415        26                     BOTTLE OF wINE
                           416       100                     wINE GLASSES
                           422         2                     SUNKEN BATH

ORDER TOTALS               1648      150                   4      ORDER VALUE   3562.10    CUSTOMER EXCEEDED CREDIT LIMIT BY 1.362.10
*****************************************************************************************
BILL TO            2  000990    MINARD STEEL FABRICATION       PP92KL                      CUSTOMER NOT ON MASTER FILE
                                1635 WEST MANCHESTER
                                ST. LOUIS,MO.

                      ITEM NUMBER    QTY      PRICE     QTY       DESCRIPTION
                                   ORDERED            SHIPPED
                           508        20                     LINE METAL FLANGE
                           196         6                     MILLING COMPOUND
                           197         8                     3000 DFM FAN
                           260         6                     SHAFT ASSEMBLY

ORDER TOTALS               1161       40                   0      ORDER VALUE   3327.70
*****************************************************************************************
BILL TO            3  000971    CHAPMAN SUPPLY                 PP6548TY

SHIP TO            3            SCHMIDT STEEL FABRICATION      SHIPPING INSTRUCTIONS
                                2509 NOTHERN HILL CT           C.O.D.
                                ATCO, N.J.              29879

                      ITEM NUMBER    QTY      PRICE     QTY       DESCRIPTION
                                   ORDERED            SHIPPED
                           197         5                     1300CFM FAN  REBUILT
                                   12000                                                   QUANTITY ORDERED OVER 1000
                                                                                           ITEM NOT ON MASTER FILE
                           415        17                     BOTTLE OF wINE

ORDER TOTALS               612     12022                   0      ORDER VALUE    778.20
*****************************************************************************************
BILL TO               000981    SATTERFIELD SPIRITS CO.        PP88117                     ORDER NUMBER BLANK

ORDER TOTALS               0           0                   0      ORDER VALUE       .00    MISSING ITEM RECORDS
*****************************************************************************************
BILL TO            4  000941    SINCLAIR DRY GOODS CO.

SHIP TO            4            O'BRIEN MFG. CO.               SHIPPING INSTRUCTIONS
                                5855 MONROE STREET
                                TOLEDO, OH                                                 ZIP CODE BLANK

                      ITEM NUMBER    QTY      PRICE     QTY       DESCRIPTION
                                   ORDERED            SHIPPED
                           508         5                     LINE METAL FLANGE



DATE  5/23/83
               ORDER   CUSTOMER   CUSTOMER                                                     ERROR MESSAGES
               NUMBER  NUMBER     NAME                         P.O. NUMBER

                           196                               MILLING COMPOUND              QUANTITY ORDERED ZERO
                           197         6                     3000 DFM FAN
                           260         6                     SHAFT ASSEMBLY
                           217         2                                                   ITEM NOT ON MASTER FILE
DU000005000901                                                                                        00000AD44496 041477
                                                                                           INVALID RECORD CODE

ORDER TOTALS               1378       19                   0      ORDER VALUE   2503.45
*****************************************************************************************
          **************************NO BILL TO**************************                  MISSING HEADER RECORD

SHIP TO            5            SCHNARE & MOLLER INC           SHIPPING INSTRUCTIONS
                                TWO PENN PLAZA                 ASAP
                                NEw YORK, NY            22508

                      ITEM NUMBER    QTY      PRICE     QTY       DESCRIPTION
                                   ORDERED            SHIPPED
                           415        20                     BOTTLE OF wINE
                           422         1                     SUNKEN BATH

ORDER TOTALS               837        21                   0      ORDER VALUE    257.50
*****************************************************************************************  ORDER NUMBER BLANK
BILL TO               000000                                                               CUSTOMER NOT ON MASTER FILE

ORDER TOTALS               0           0                   0      ORDER VALUE       .00    MISSING ITEM RECORDS
*****************************************************************************************


                      ITEM NUMBER      QTY ORDERED                 QTY SHIPPED
FINAL TOTALS               5636          12252                         4
```

BEST COPY AVAILABLE

Part # 3 Con't : Edit Report II

```
DATE  5/23/83
          ORDER    CUSTOMER   CUSTOMER                                                 ERROR MESSAGES
          NUMBER   NUMBER     NAME                       P.O. NUMBER

BILL TO          1  000941    SATTERFIELD SPIRITS CO.       PP88117

SHIP TO          1            LAMKIN & ASSOCIATES          SHIPPING INSTRUCTIONS
                              CANYON ESTATES               C.O.D.
                              CARROLLTON,MO.        19768

                  ITEM NUMBER     QTY     PRICE     QTY       DESCRIPTION
                                ORDERED           SHIPPED
                        185        2                         CUP DISPENSER
                        229       16                         GLASS JAR 1 QT.
                                   4                     4   METAL CAN 1 GALLON         ITEM NOT ON MASTER FILE
                        415       26                         BOTTLE OF WINE
                        416      100                         WINE GLASSES
                        422        2                         SUNKEN BATH

ORDER TOTALS           1648      150                     4       ORDER VALUE   3562.10  CUSTOMER EXCEEDED CREDIT LIMIT BY 1.362.10
**************************************************************************************
BILL TO          2  000990    MINARD STEEL FABRICATION      PP92KL                     CUSTOMER NOT ON MASTER FILE
                              1835 WEST MANCHESTER
                              ST. LOUIS,MO.

                  ITEM NUMBER     QTY     PRICE     QTY       DESCRIPTION
                                ORDERED           SHIPPED
                        508       20                         LINE METAL FLANGE
                        196        8                         MILLING COMPOUND
                        197        8                         3000 DFM FAN
                        260        6                         SHAFT ASSEMBLY

ORDER TOTALS           1161       40                     0       ORDER VALUE   3327.70
**************************************************************************************
BILL TO          3  000971    CHAPMAN SUPPLY                PP6548TY

SHIP TO          3            SCHMIDT STEEL FABRICATION    SHIPPING INSTRUCTIONS
                              2539 NOTHERN HILL CT         C.O.D.
                              ATCO,N.J.             29879

                  ITEM NUMBER     QTY     PRICE     QTY       DESCRIPTION
                                ORDERED           SHIPPED
                        197        5                         1300CFM FAN  REBUILT
                               12000                                                    QUANTITY ORDERED OVER 1000
                                                                                        ITEM NOT ON MASTER FILE
                        415       17                         BOTTLE OF WINE

ORDER TOTALS            612    12022                     0       ORDER VALUE    778.20
**************************************************************************************
BILL TO             000981    SATTERFIELD SPIRITS CO.       PP88117                    ORDER NUMBER BLANK

ORDER TOTALS              0        8                     0       ORDER VALUE       .00  MISSING ITEM RECORDS
**************************************************************************************
BILL TO          4  000941    SINCLAIR DRY GOODS CO.

SHIP TO          4            O'BRIEN MFG. CO.             SHIPPING INSTRUCTIONS
                              5855 MONROE STREET
                              TOLEDO,OH                                                 ZIP CODE BLANK

                  ITEM NUMBER     QTY     PRICE     QTY       DESCRIPTION
                                ORDERED           SHIPPED
                        508        5                         LINE METAL FLANGE


DATE  5/23/83
          ORDER    CUSTOMER   CUSTOMER                                                 ERROR MESSAGES
          NUMBER   NUMBER     NAME                       P.O. NUMBER

                        196                                  MILLING COMPOUND           QUANTITY ORDERED ZERO
                        197        6                         3000 DFM FAN
                        260        6                         SHAFT ASSEMBLY
                        217        2                                                    ITEM NOT ON MASTER FILE
0U000005000901                                                                                     00000AD44496 041477
                                                                                        INVALID RECORD CODE

ORDER TOTALS           1378       13                     0       ORDER VALUE   2503.45
**************************************************************************************
          ************************NO BILL TO************************                   MISSING HEADER RECORD

SHIP TO          5            SCHNARE & MOLLER INC         SHIPPING INSTRUCTIONS
                              TWO PENN PLAZA               ASAP
                              NEW YORK, NY          22508

                  ITEM NUMBER     QTY     PRICE     QTY       DESCRIPTION
                                ORDERED           SHIPPED
                        415       20                         BOTTLE OF WINE
                        422        1                         SUNKEN BATH

ORDER TOTALS            837       21                     0       ORDER VALUE    257.50
**************************************************************************************
BILL TO             000000                                                              ORDER NUMBER BLANK
                                                                                        CUSTOMER NOT ON MASTER FILE

ORDER TOTALS              0        0                     0       ORDER VALUE       .00  MISSING ITEM RECORDS
**************************************************************************************

                  ITEM NUMBER         QTY ORDERED              QTY SHIPPED
FINAL TOTALS             5636               12252
```

DATA ENTRY AND EDIT PROGRAMS

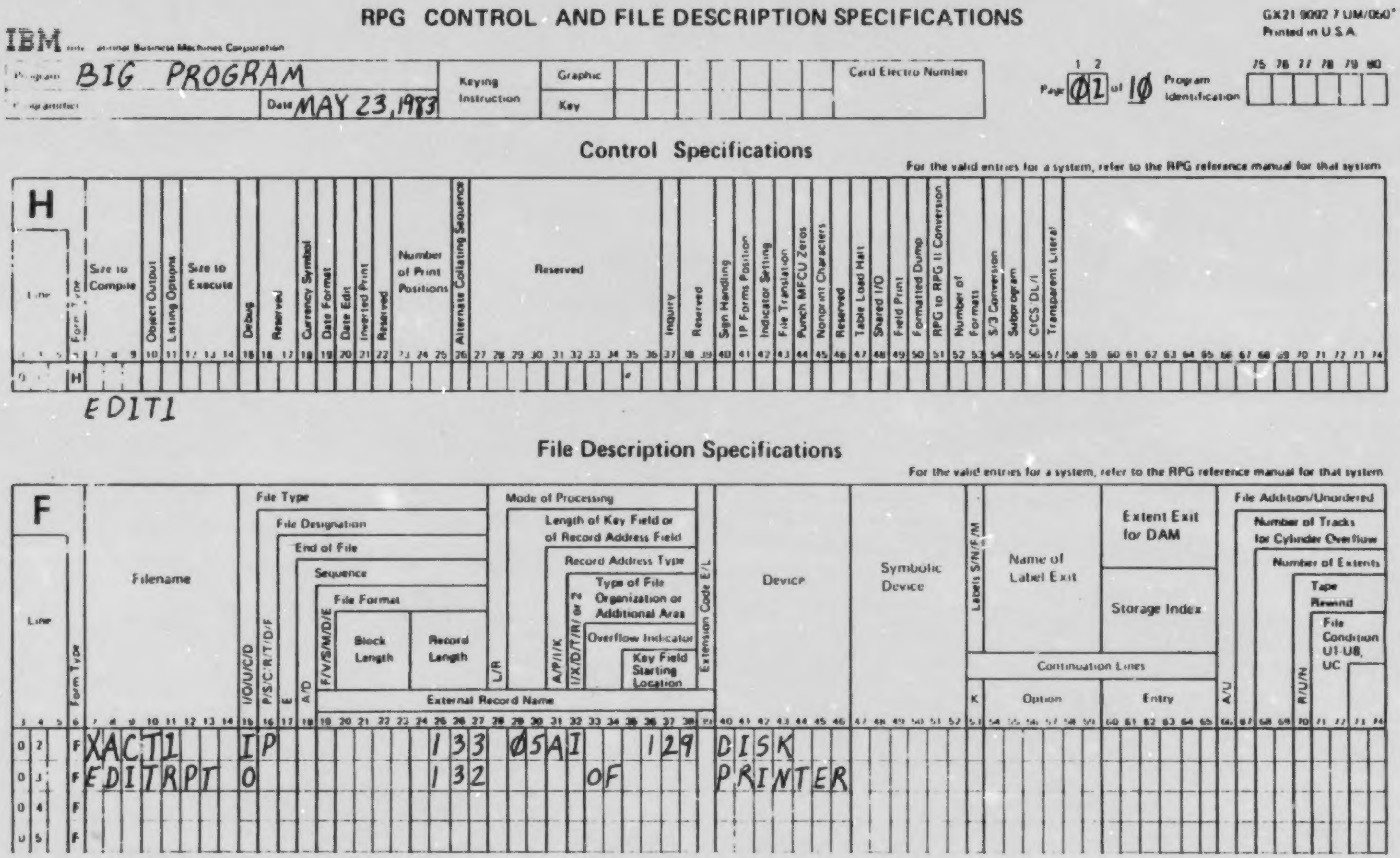
UNIT XVII

ANSWERS TO ASSIGNMENT SHEETS

ASSIGNMENT SHEET 1 and 2

(see following pages)

Assignment Sheet 1: Source Program Solution for Problem 1



IBM International Business Machines Corporation

RPG CONTROL AND FILE DESCRIPTION SPECIFICATIONS

GX21 9092 7 UM/050*
Printed in U.S.A.

Program: BIG PROGRAM
Programmer:
Date: MAY 23, 1983
Keying Instruction: Graphic / Key
Card Electro Number
Page 01 of 10
Program Identification (75–80)

Control Specifications

For the valid entries for a system, refer to the RPG reference manual for that system

Line	Form Type	Size to Compile	Object Output	Listing Options	Size to Execute	Debug	Reserved	Currency Symbol	Date Format	Date Edit	Inverted Print	Reserved	Number of Print Positions	Alternate Collating Sequence	Reserved	Inquiry	Reserved	Sign Handling	1P Forms Position	Indicator Setting	File Translation	Punch MFCU Zeros	Nonprint Characters	Reserved	Table Load Halt	Shared I/O	Field Print	Formatted Dump	RPG to RPG II Conversion	Number of Formats	S/3 Conversion	Subprogram	CICS DL/I	Transparent Literal
0	H																																	

EDIT1

File Description Specifications

For the valid entries for a system, refer to the RPG reference manual for that system

Line	Form Type	Filename	File Type (I/O/U/C/D)	File Designation (P/S/C/R/T/D/F)	End of File (E)	Sequence (A/D)	File Format (F/V/S/M/D/E)	Block Length	Record Length	L/R	Length of Key Field or of Record Address Field	Record Address Type (A/P/I/K)	Type of File Organization or Additional Area (I/X/D/T/R/ or 2)	Overflow Indicator	Key Field Starting Location	Extension Code E/L	Device	Symbolic Device	Labels S/N/E/M	Name of Label Exit	Extent Exit for DAM	A/U	File Addition/Unordered	Number of Tracks for Cylinder Overflow	Number of Extents	Tape Rewind	File Condition U1-U8, UC
02	F	XACT1	I	P					133		05	A	I		129		DISK										
03	F	EDITRPT	O						132					OF			PRINTER										
04	F																										
05	F																										

BEST COPY AVAILABLE

Assignment Sheet 1: Source Program Solution for Problem 1

IBM International Business Machines Corporation

RPG INPUT SPECIFICATIONS

GX21-9094-4 UM/050
Printed in U.S.A.

Program		Keying Instruction	Graphic	Card Electro Number	Page 02 of 10
Programmer	Date		Key		Program Identification 75 76 77 78 79

Line	Form Type	Filename or Record Name	Sequence	Number (1/N), E	Option (O), U, S	Record Identifying Indicator, ** or DS	1 Position	1 Not (N)	1 C/Z/D	1 Character	2 Position	2 Not (N)	2 C/Z/D	2 Character	3 Position	3 Not (N)	3 C/Z/D	3 Character	Stacker Select	P/B/L/R	Field Location From	Field Location To	Decimal Positions	RPG Field Name	Control Level (L1-L9)	Matching Fields or Chaining Fields	Field Record Relation	Field Indicators Plus	Minus	Zero or Blank
01	I	XACT1	NS			01	1		C	C	2		C	U																
02	I																				1	2		RCDCD						
03	I																				3	8	0	ORDR#	L1					
04	I																				9	14	0	CUST#						
05	I																				15	39		CUSNM						
06	I																				40	64		ADD1						
07	I																				65	89		ADD2						
08	I																				90	109		ADD3						
09	I																				110	114	0	ZIPCD						
10	I																				115	122		PO#						
11	I																				123	128	0	ODATE						
12	I																				129	130		BLANK						
13	I		NS			02	1		C	S	2		C	H																
14	I																				1	2		RCDCD						
15	I																				3	8	0	ORDR#	L1					
16	I																				9	33		SNAME						
17	I																				34	58		SADD1						
18	I																				59	83		SADD2						
19	I																				84	103		SINST						
20	I																				104	108	0	ZIPCD						

BEST COPY AVAILABLE

Assignment Sheet 1: Source Program Solution for Program 1

IBM International Business Machines Corporation

RPG INPUT SPECIFICATIONS

GX21-9094-4 UM/050*
Printed in U.S.A.

Program
Programmer
Date
Keying Instruction: Graphic / Key
Card Electro Number
Page 03 of 10
Program Identification (75 76 77 78 79 80)

I

Line (3-5)	Form Type (6)	Filename or Record Name / Data Structure Name (7-14)	Sequence (15-16)	Number (1/N), E (17)	Option (O), U, S (18)	Record Identifying Indicator, ** or DS (19-20)	1 Position (21-24)	1 Not (N) (25)	1 C/Z/D (26)	1 Character (27)	2 Position (28-31)	2 Not (N) (32)	2 C/Z/D (33)	2 Character (34)	3 Position (35-38)	3 Not (N) (39)	3 C/Z/D (40)	3 Character (41)	Stacker Select (42)	P/B/L/R (43)	Field Location From / Data Structure Occurs n Times (44-47)	Field Location To / Length (48-51)	Decimal Positions (52)	RPG Field Name (53-58)	Control Level (L1-L9) (59-60)	Matching Fields or Chaining Fields (61-62)	Field Record Relation (63-64)	Field Indicators Plus (65-66)	Field Indicators Minus (67-68)	Field Indicators Zero or Blank (69-70)	(71-74)
01	I		NS			03	1		C	I	2		C	T																	
02	I																				1	2		RCDCD							
03	I																				3	8	0	ORDR#	L1						
04	I																				9	16	0	ITEM#							
05	I																				32	36	0	QTYO							
06	I																				37	41	2	PRICE							
07	I																				56	60	0	QTYS							
08	I																				61	87		DESC							
09	I		NS			04																									
10	I																				1	2		RCDCD							
11	I																				3	8	0	ORDR#							
12	I																				9	14	0	CUST#							
13	I																				110	114	0	ZIPCD							
14	I																				115	122		PO#							
15	I																				123	128	0	ODATE							
16	I																														
17	I																														
18	I																														
19	I																														
20	I																														

BEST COPY AVAILABLE

Assignment Sheet 1: Source Program Solution for Problem 1

IBM International Business Machines Corporation

RPG CALCULATION SPECIFICATIONS

GX21 9093-3 UM 050
Printed in U.S.A.

Program
Programmer
Date
Keying Instruction: Graphic / Key
Card Electro Number
Page 04 of 10
Program Identification

Line	Form Type	Control Level (L0-L9, LR, SR, AN/OR)	Indicators	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Half Adjust (H)	Resulting Indicators High / Plus / 1 > 2	Resulting Indicators Low / Minus / 1 < 2	Resulting Indicators Equal / Zero / 1 = 2	Comments
01	C				SETOF						11	12	13	
02	C				SETOF						14	20	21	
03	C				SETOF						23	30	31	
04	C				SETOF						32	50	25	
05	C				SETON						26			
06	C		01		SETON						11			
07	C		02		SETON						12			
08	C		03		SETON						13			
09	C		04		SETON						14			
10	C		14		GOTO	END								
11	C		11		SETON						39			
12	C		11		SETOF						38			
13	C		11		Z-ADD	0	COUNT	2	0					
14	C		11	ORDR#	COMP	0							20	
15	C		11	CUSNM	COMP	BLANK					21	21	25	
16	C		12		Z-ADD	0	COUNT	2	0					
17	C		12	ZIPCD	COMP	0							23	
18	C		12		GOTO	END								
19	C		13		SETOF						39	26		
20	C		13		SETON						38			

BEST COPY AVAILABLE

Assignment Sheet 1: Source Program Solution for Problem 1

IBM International Business Machines Corporation

RPG CALCULATION SPECIFICATIONS

GX21 9093 3 UM 050
Printed in U.S.A.

Program
Programmer
Date
Keying Instruction: Graphic / Key
Card Electro Number
Page 05 of 10
Program Identification (75–80)

Line (3–5)	Form Type (6)	Control Level (7–8)	Indicators (9–17)	Factor 1 (18–27)	Operation (28–32)	Factor 2 (33–42)	Result Field Name (43–48)	Length (49–51)	Decimal Positions (52)	Half Adjust (53)	Resulting Indicators High / Plus / 1>2 (54–55)	Resulting Indicators Low / Minus / 1<2 (56–57)	Resulting Indicators Equal / Zero / 1=2 (58–59)	Comments (60–74)
01	C		13	ITEM#	COMP	0							30	
02	C		13	QTYO	COMP	0							31	
03	C		13	QTYO	COMP	1000					32			
04	C		13	ITEMT	ADD	ITEM#	ITEMT	4	0					
05	C		13	QTYOT	ADD	QTYO	QTYOT	5	0					
06	C		13	PRICET	ADD	PRICE	PRICET	5	2					
07	C		13	QTYST	ADD	QTYS	QTYST	5	0					
08	C		13	TOT1	ADD	ITEM#	TOT1	4	0					
09	C		13	TOT2	ADD	QTYO	TOT2	5	0					
10	C		13	TOT3	ADD	PRICE	TOT3	5	2					
11	C		13	TOT4	ADD	QTYS	TOT4	5	0					
12	C		13	COUNT	COMP	0							50	
13	C		13	COUNT	ADD	1	COUNT	2	0					
14	C			END	TAG									
15	C													
16	C													
17	C													
18	C													
19	C													
20	C													

BEST COPY AVAILABLE

Assignment Sheet 1: Source Program Solution for Problem 1

IBM International Business Machines Corporation

RPG OUTPUT SPECIFICATIONS

GX21-9090-4 UM/050*
Printed in U.S.A.

Program | Programmer | Date | Keying Instruction | Graphic | Key | Card Electro Number

Page 06 of 10 | Program Identification

```
01 OEDITRPT H  202   1P
02 O       OR        OF
03 O                                    4 'DATE'
04 O                         UDATE Y   13
05 O                                   66 'EDIT REPORT 1'
06 O                                  123 'PAGE'
07 O                         PAGE  Z  128
08 O        H  1     1P
09 O       OR        OF
10 O                                   16 'ORDER'
11 O                                   27 'CUSTOMER'
12 O                                   38 'CUSTOMER'
13 O                                  109 'ERROR MESSAGES'
14 O        H  2     1P
15 O       OR        OF
16 O                                   17 'NUMBER'
17 O                                   25 'NUMBER'
18 O                                   34 'NAME'
19 O                                   72 'P.O. NUMBER'
20 O        D  1     14
```

BEST COPY AVAILABLE

Assignment Sheet 1: Source Program Solution for Problem 1

IBM International Business Machines Corporation

RPG OUTPUT SPECIFICATIONS

GX21-9090-4 UM/050*
Printed in U.S.A.

Program		Keying Instruction	Graphic
Programmer	Date		Key

Card Electro Number

Page 07 of 10

Program Identification (75 76 77 78 79 80)

Commas	Zero Balances to Print	No Sign	CR	–
Yes	Yes	1	A	J
Yes	No	2	B	K
No	Yes	3	C	L
No	No	4	D	M

X = Remove Plus Sign
Y = Date Field Edit
Z = Zero Suppress

5 - 9 = User Defined

Line (3-5)	Form Type (6)	Filename or Record Name (7-14)	Type (H/D/T/E) (15)	Space Before (17)	Space After (18)	Skip Before (19-20)	Skip After (21-22)	Output Indicators Not (23)	Output Indicators (24-25)	And Not (26)	And (27-28)	And Not (29)	And (30-31)	Field Name or EXCPT Name (32-37)	Edit Codes (38)	B/A/C/1-9/R (39)	End Position in Output Record (40-43)	P/B/L/R (44)	Constant or Edit Word (45-70)
01	O													RCDCD			2		
02	O													ORDR#			8		
03	O													CUST#			14		
04	O													ZIPCD			114		
05	O													PO#			122		
06	O													ODATE			128		
07	O		D		1				14										
08	O																109		'INVALID RECORD CODE'
09	O		D		0				11										
10	O																7		'BILL TO'
11	O													ORDR#	Z		17		
12	O													CUST#			25		
13	O													CUSNM			54		
14	O													PO#			69		
15	O								11		20						108		'ORDER NUMBER BLANK'
16	O		D	1	0			N	25		21								
17	O													ADD1			54		
18	O		D	1	0			N	25		21								
19	O													ADD2			54		
20	O		D		2				12		38								

BEST COPY AVAILABLE

Assignment Sheet 1: Source Program Solution for Problem 1

IBM International Business Machines Corporation

RPG OUTPUT SPECIFICATIONS

GX21-9090-4 UM/050*
Printed in U.S.A.

Program		Keying Instruction	Graphic
Programmer	Date		Key

Card Electro Number

Page 08 of 10 Program Identification

Edit Codes	Commas	Zero Balances to Print	No Sign	CR	–
	Yes	Yes	1	A	J
	Yes	No	2	B	K
	No	Yes	3	C	L
	No	No	4	D	M

X = Remove Plus Sign
Y = Date Field Edit
Z = Zero Suppress

5-9 = User Defined

Line	Form Type	Filename or Record Name	Type (H/D/T/E)	Space Before	Space After	Output Indicators	And	Field Name or EXCPT Name	Edit Codes	End Position in Output Record	Constant or Edit Word
01	O									36	'*————————*'
02	O									46	'NO BILL TO'
03	O									70	'*————————*'
04	O									111	'MISSING HEADER RECORD'
05	O		D		1	12					
06	O									7	'SHIP TO'
07	O							ORDR#	2	17	
08	O							SNAME		54	
09	O									81	'SHIPPING INSTRUCTIONS'
10	O		D		1	12					
11	O							SADD1		54	
12	O							SINST		80	
13	O		D		0	12					
14	O							SADD2		54	
15	O							ZIPCD	2	58	
16	O					12	23			104	'ZIP CODE BLANK'
17	O		D	2	1	13	50				
18	O									32	'ITEM NUMBER'
19	O									39	'QTY'
20	O									49	'PRICE'

BEST COPY AVAILABLE

Assignment Sheet 1: Source Program Solution for Problem 1

IBM International Business Machines Corporation

RPG OUTPUT SPECIFICATIONS

GX21-9090-4 U
Printed in U.S.A

Program
Programmer
Date
Keying Instruction: Graphic / Key
Card Electro Number
Page 09 of 10
Program Identification 75 76 77 78

Edit code key (Constant or Edit Word):

Commas	Zero Balances to Print	No Sign	CR	–
Yes	Yes	1	A	J
Yes	No	2	B	K
No	Yes	3	C	L
No	No	4	D	M

X = Remove Plus Sign
Y = Date Field Edit
Z = Zero Suppress
5-9 = User Defined

Line (3-5)	Form Type (6)	Filename or Record Name (7-14)	Type (H/D/T/E) (15)	Space Before (17)	Space After (18)	Skip Before (19-20)	Skip After (21-22)	Output Indicators (23-25)	And (26-28)	And (29-31)	Field Name or EXCPT Name (32-37)	Edit Codes (38)	End Position in Output Record (40-43)	Constant or Edit Word (45-70)
01	O												57	'QTY'
02	O												74	'DESCRIPTION'
03	O		D		1			13	50					
04	O												41	'ORDERED'
05	O												59	'SHIPPED'
06	O		D		1			13						
07	O										ITEM#	Z	30	
08	O										QTYO	Z	40	
09	O										PRICE	1	49	
10	O										QTYS	Z	58	
11	O										DESC		86	
12	O							13	30				109	'ITEM NUMBER MISSING'
13	O							13	31				111	'QUANTITY ORDERED ZERO'
14	O		D		1			13	32					
15	O							13	32				112	'QUANTITY ORDERED OVER'
16	O							13	32				116	'1000'
17	O		T		1			L1	N26					
18	O												12	' '
19	O		T		1			L1						
20	O												12	'ORDER TOTALS'

BEST COPY AVAILABLE

Assignment Sheet 1: Source Program Solution for Problem 1

IBM International Business Machines Corporation

RPG OUTPUT SPECIFICATIONS

GX21-9090-4 UM/050*
Printed in U.S.A.

Program		Keying Instruction	Graphic
Programmer	Date		Key

Card Electro Number

Page 1 0 of 10

Program Identification (75 76 77 78 79 80)

Commas	Zero Balances to Print	No Sign	CR	-
Yes	Yes	1	A	J
Yes	No	2	B	K
No	Yes	3	C	L
No	No	4	D	M

X = Remove Plus Sign
Y = Date Field Edit
Z = Zero Suppress

5-9 = User Defined

Line (3-5)	Form Type (6)	Filename or Record Name (7-14)	Type (H/D/T/E) (15)	Space Before (17)	Space After (18)	Skip Before (19-20)	Skip After (21-22)	Output Indicators (23-31)	Field Name or EXCPT Name (32-37)	Edit Codes (38)	B/A/C/1-9/R (39)	End Position in Output Record (40-43)	P/B/L/R (44)	Constant or Edit Word (45-70)
01	O								ITEMT	3	B	30		
02	O								QTYOT	3	B	40		
03	O								PRICET	1	B	49		
04	O								QTYST	3	B	58		
05	O											110		'MISSING ITEM RECORDS'
06	O		T		1			39						
07	O							L1				22		'*————————————————————*'
08	O											44		'*————————————————————*'
09	O											66		'*————————————————————*'
10	O											86		'*——————————————————*'
11	O		T	2	1			LR						
12	O											31		'ITEM NUMBER'
13	O											49		'QTY ORDERED'
14	O											59		'PRICE'
15	O											78		'QTY SHIPPED'
16	O		T					LR						
17	O											12		'FINAL TOTALS'
18	O								TOT1	Z		31		
19	O								TOT2	Z		48		
20	O								TOT3	1		59		
	O								TOT4	Z		76		
	O													

BEST COPY AVAILABLE

Assignment Sheet 2:
Source Program Solution for Problem 2

RPG CONTROL AND FILE DESCRIPTION SPECIFICATIONS

IBM International Business Machines Corporation

GX21 9092 7 UM/050*
Printed in U.S.A.

Program ORDER ENTRY EDIT II
Programmer LAMKIN
Date 5/31/83

Keying Instruction | Graphic | Key

Card Electro Number

Page 01 of 11

Program Identification EDIT2

Control Specifications

For the valid entries for a system, refer to the RPG reference manual for that system

Line	Form Type
01	H

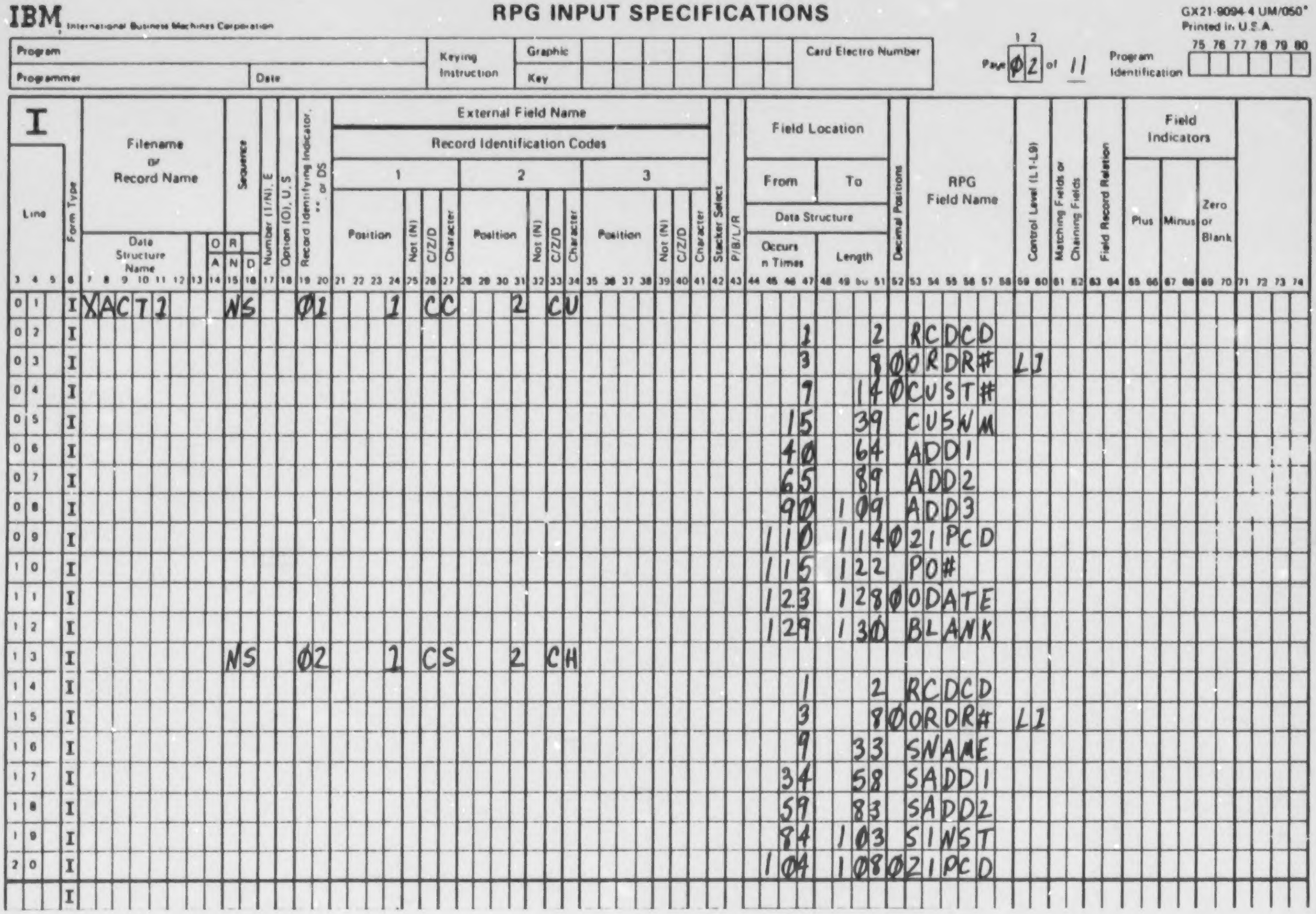
File Description Specifications

For the valid entries for a system, refer to the RPG reference manual for that system

Line	Form Type	Filename	File Type	File Designation	Record Length	Mode of Processing	Length of Key Field or of Record Address Field	Record Address Type	Type of File Organization or Additional Area	Overflow Indicator	Key Field Starting Location	Device
02	F	XACT1	I	P	133							DISK
03	F	CMAST	I	C	256	R	Ø6	A	I		3	DISK
04	F	ITEMMST	I	C		R	Ø8	A	I		4	DISK
05	F	EDITRPT	O		132					OF		PRINTER
06	F											

BEST COPY AVAILABLE

Assignment Sheet 2: Source Program Solution for Problem 2



IBM International Business Machines Corporation

RPG INPUT SPECIFICATIONS

GX21-9094-4 UM/050*
Printed in U.S.A.

Program
Programmer
Date
Keying Instruction
Graphic
Key
Card Electro Number
Page 02 of 11
Program Identification

Line	Form Type	Filename or Record Name	Sequence	Record Identifying Indicator	Position (1)	C/Z/D (1)	Character (1)	Position (2)	C/Z/D (2)	Character (2)	Field Location From	Field Location To	Decimal Positions	RPG Field Name	Control Level (L1-L9)
01	I	XACT1	NS	01	1	C	C	2	C	U					
02	I										1	2		RCDCD	
03	I										3	8	0	ORDR#	L1
04	I										9	14	0	CUST#	
05	I										15	39		CUSNM	
06	I										40	64		ADD1	
07	I										65	89		ADD2	
08	I										90	109		ADD3	
09	I										110	114	0	21PCD	
10	I										115	122		PO#	
11	I										123	128	0	ODATE	
12	I										129	130		BLANK	
13	I		NS	02	1	C	S	2	C	H					
14	I										1	2		RCDCD	
15	I										3	8	0	ORDR#	L1
16	I										9	33		SNAME	
17	I										34	58		SADD1	
18	I										59	83		SADD2	
19	I										84	103		SINST	
20	I										104	108	0	21PCD	
	I														

BEST COPY AVAILABLE

Assignment Sheet 2: Source Program Solution for Problem 2

IBM International Business Machines Corporation

RPG INPUT SPECIFICATIONS

GX21 9094 4 UM/050
Printed in U.S.A.

Program
Programmer
Date
Keying Instruction
Graphic
Key
Card Electro Number
Page 03 of 11
Program Identification

Line	Form Type	Filename or Record Name	Sequence	Record Identifying Indicator	Record Identification Codes 1: Position	1: C/Z/D	1: Character	2: Position	2: C/Z/D	2: Character	Field Location From	Field Location To	Decimal Positions	RPG Field Name	Control Level (L1-L9)
01	I		NS	03	1	C	I	2	C	T					
02	I										1	2		RCDCD	
03	I										3	8	0	ORDR#	L1
04	I										9	16	0	ITEM#	
05	I										32	36	0	QTYO	
06	I										37	41	2	PRICE	
07	I										56	60	0	QTYS	
08	I										61	87		DESC	
09	I		NS	04											
10	I										1	2		RCDCD	
11	I										3	8	0	ORDR#	
12	I										9	14	0	CUST#	
13	I										110	114	0	ZIPCD	
14	I										115	122		PO#	
15	I										123	128	0	ODATE	
16	I	CMAST	NS	05	1	C	M	2	C	A					
17	I										1	2		RCDCD	
18	I										3	8		CUST#	
19	I										9	33		CUSNM	
20	I										136	140	0	CLAMT	
	I										171	177	2	BAL	

BEST COPY AVAILABLE

Assignment Sheet 2: Source Program Solution for Problem 2

IBM International Business Machines Corporation

RPG INPUT SPECIFICATIONS

GX21-9094-4 UM/050*
Printed in U.S.A.

Program
Programmer
Date
Keying Instruction: Graphic, Key
Card Electro Number
Page 04 of 11
Program Identification (75–80)

Line (3–5)	Form Type (6)	Filename or Record Name (7–14)	Sequence (15–16)	Number (1/N), E (17)	Option (O), U, S (18)	Record Identifying Indicator, ** or DS (19–20)	1 Position (21–24)	1 Not (N) (25)	1 C/Z/D (26)	1 Character (27)	2 Position (28–31)	2 Not (N) (32)	2 C/Z/D (33)	2 Character (34)	3 Position (35–38)	3 Not (N) (39)	3 C/Z/D (40)	3 Character (41)	Stacker Select (42)	P/B/L/R (43)	Field Location From (44–47)	Field Location To (48–51)	Decimal Positions (52)	RPG Field Name (53–58)	Control Level (L1-L9) (59–60)	Matching Fields or Chaining Fields (61–62)	Field Record Relation (63–64)	Field Indicators Plus (65–66)	Field Indicators Minus (67–68)	Field Indicators Zero or Blank (69–70)	(71–74)
01	I	ITEMMST	NS			06	1		C	I	2		C	M																	
02	I																				1	2		RCDCD							
03	I																				4	11	0	ITEM#							
04	I																				32	58		DESC							
05	I																				77	81	2	PRICE							
06	I																														
07	I																														
08	I																														
09	I																														
10	I																														
11	I																														
12	I																														
13	I																														
14	I																														
15	I																														
16	I																														
17	I																														
18	I																														
19	I																														
20	I																														

BEST COPY AVAILABLE

Assignment Sheet 2: Source Program Solution for Problem 2

IBM International Business Machines Corporation

RPG CALCULATION SPECIFICATIONS

GX21 9093 3 UM 050*
Printed in U.S.A.

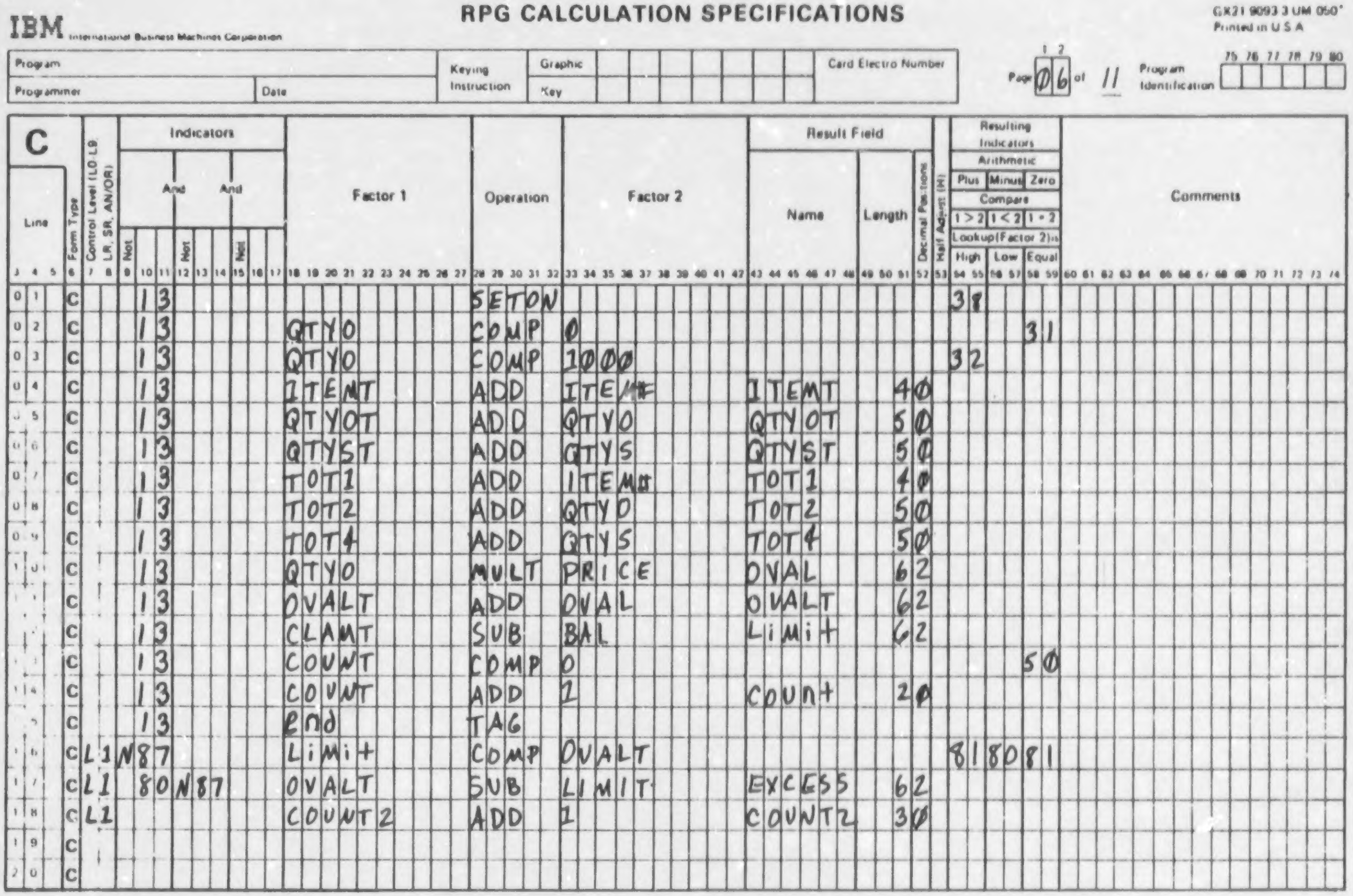
Program
Programmer
Date
Keying Instruction: Graphic / Key
Card Electro Number
Page 05 of 11
Program Identification

Line	Form Type	Control Level	Indicators	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Half Adjust	Resulting Indicators High (Plus, 1>2)	Low (Minus, 1<2)	Equal (Zero, 1=2)	Comments
01	C				SETOF						85	81		
02	C				SETOF						60	70	80	
03	C				SETOF						11	12	13	
04	C				SETOF						14	20	21	
05	C				SETOF						23	30	31	
06	C				SETOF						32	50	25	
07	C				SETON						26			
08	C		03	PRICE	COMP	145.95							85	
09	C		01	CUST#	CHAIN	CMAST					60			
10	C		03 N85	ITEM#	CHAIN	ITEMMST					70			
11	C		01	COUNT2	SETON						11			
12	C		02		SETON						12			
13	C		03		SETON						13			
14	C		04		SETON						14			
15	C		14		GOTO	END								
16	C		11		COMP	1							87	
17	C		11		SETON						39			
18	C		11		SETOF						38			
19	C		11		Z-ADD	0	COUNT	2	0					
20	C		11	ORDR#	COMP	0							20	
	C		11	CUSNM	COMP	BLANK					21	21	25	
	C		12		Z-ADD	0	COUNT	2	0					
	C		12	ZIPCD	COMP	0							23	
	C		12		GOTO	END								
	C		13		SETOF						39	26		

*Number of sheets per pad may vary slightly.

BEST COPY AVAILABLE

Assignment Sheet 2: Source Program Solution for Problem 2



IBM International Business Machines Corporation

RPG CALCULATION SPECIFICATIONS

GX21 9093 3 UM 050*
Printed in U.S.A.

Program		Keying Instruction	Graphic
Programmer	Date		Key

Card Electro Number

Page 06 of 11

Program Identification

Line	Form Type	Control Level (L0-L9, LR, SR, AN/OR)	Indicators	And	And	Factor 1	Operation	Factor 2	Result Field Name	Length	Decimal Positions	Half Adjust (H)	Resulting Indicators: Plus / 1>2 / High	Minus / 1<2 / Low	Zero / 1=2 / Equal	Comments
01	C		13				SETON						38			
02	C		13			QTYO	COMP	0							31	
03	C		13			QTYO	COMP	1000					32			
04	C		13			ITEMT	ADD	ITEM#	ITEMT	4	0					
05	C		13			QTYOT	ADD	QTYO	QTYOT	5	0					
06	C		13			QTYST	ADD	QTYS	QTYST	5	0					
07	C		13			TOT1	ADD	ITEM#	TOT1	4	0					
08	C		13			TOT2	ADD	QTYO	TOT2	5	0					
09	C		13			TOT4	ADD	QTYS	TOT4	5	0					
10	C		13			QTYO	MULT	PRICE	OVAL	6	2					
11	C		13			OVALT	ADD	OVAL	OVALT	6	2					
12	C		13			CLAMT	SUB	BAL	Limit	6	2					
13	C		13			COUNT	COMP	0							50	
14	C		13			COUNT	ADD	1	Count	2	0					
15	C		13			end	TAG									
16	C	L1	N87			Limit	COMP	OVALT					81	80	81	
17	C	L1	80	N87		OVALT	SUB	LIMIT	EXCESS	6	2					
18	C	L1				COUNT2	ADD	1	COUNT2	3	0					
19	C															
20	C															

BEST COPY AVAILABLE

Assignment Sheet 2: Source Program Solution for Problem 2

IBM International Business Machines Corporation

RPG OUTPUT SPECIFICATIONS

GX21-9090-4 UM/050*
Printed in U.S.A.

Program
Programmer
Date
Keying Instruction: Graphic / Key
Card Electro Number
Page 07 of 11
Program Identification

Commas	Zero Balances to Print	No Sign	CR	–
Yes	Yes	1	A	J
Yes	No	2	B	K
No	Yes	3	C	L
No	No	4	D	M

X = Remove Plus Sign
Y = Date Field Edit
Z = Zero Suppress
5-9 = User Defined

```
  01 OEDITRPT H  102   1P
  02 O       OR        OF
  03 O                                    4 'DATE'
  04 O                         UDATE Y   13
  05 O                                   66 'EDIT REPORT 1'
  06 O                                  123 'PAGE'
  07 O                         PAGE  Z  128
  08 O        H  1     1P
  09 O       OR        OF
  10 O                                   16 'ORDER'
  11 O                                   27 'CUSTOMER'
  12 O                                   38 'CUSTOMER'
  13 O                                  109 'ERROR MESSAGES'
  14 O        H  2     1P
  15 O       OR        OF
  16 O                                   17 'NUMBER'
  17 O                                   25 'NUMBER'
  18 O                                   34 'NAME'
  19 O                                   72 'P.O. NUMBER'
  20 O        D  1     14
     O                         RCDCD      2
     O                         ORDR#      8
     O                         CUST#     14
     O                         ZIPCD    114
     O                         PO#      122
```

*Number of sheets per pad may vary slightly

BEST COPY AVAILABLE

Assignment Sheet 2: Source Program Solution for Problem 2

IBM International Business Machines Corporation

RPG OUTPUT SPECIFICATIONS

GX21-9090-4 UM/050*
Printed in U.S.A.

Program
Programmer
Date
Keying Instruction: Graphic / Key
Card Electro Number
Page 08 of 11
Program Identification (75 76 77 78 79 80)

Commas	Zero Balances to Print	No Sign	CR	–
Yes	Yes	1	A	J
Yes	No	2	B	K
No	Yes	3	C	L
No	No	4	D	M

X = Remove Plus Sign
Y = Date Field Edit
Z = Zero Suppress
5-9 = User Defined

Line	Form Type	Filename or Record Name	Type (H/D/T/E)	Space Before	Space After	Skip Before	Skip After	Not	Output Indicator	Not	And	Not	And	Field Name or EXCPT Name	Edit Codes	Blank After	End Position in Output Record	P/B/L/R	Constant or Edit Word
01	O													ODATE			128		
02	O		D		2				14										
03	O																109		'INVALID RECORD CODE'
04	O		D		0				11										
05	O																7		'BILL TO'
06	O													ORDR#	Z		17		
07	O													CUST#			25		
08	O													CUSNM			54		
09	O													PO#			69		
10	O																		
11	O								11		20						108		'ORDER NUMBER BLANK'
12	O								11		60						113		'CUSTOMER NOT ON MASTER'
13	O								11		60						117		'FILE'
14	O		D	1	1			N	25		21								
15	O																		
16	O													ADD1			54		
17	O								11		60		20				113		'CUSTOMER NOT ON MASTER'
18	O								11		60		20				117		'FILE'
19	O		D	0	0			N	25		21								
20	O													ADD2			54		
	O		D		1				11		60		20						
	O								11		60		20				12		' '
	O		D		2				12		38								
	O																36		'************************'
	O																46		'NO BILL TO'

*Number of sheets per pad may vary slightly

BEST COPY AVAILABLE

Assignment Sheet 2: Source Program Solution for Problem 2

IBM International Business Machines Corporation

RPG OUTPUT SPECIFICATIONS

GX21-9090-4 UM/050*
Printed in U.S.A.

Program
Programmer
Date
Keying Instruction: Graphic / Key
Card Electro Number
Page 09 of 11
Program Identification

```
01 O                                   70 '************************'
02 O                                  111 'MISSING HEADER RECORD'
03 O         D 1  11
04 O                                    7 'SHIP TO'
05 O                         ORDR#     17
06 O                         SNAME     54
07 O                                   81 'SHIPPING INSTRUCTIONS'
08 O         D 1  12
09 O                         SADD1     54
10 O                         SINST     80
11 O         D 0  12
12 O                         SADD2     54
13 O                         ZIPCD Z   58
14 O              12 23
15 O         D21  13 50
16 O                                   32 'ITEM NUMBER'
17 O                                   39 'QTY'
18 O                                   49 'PRICE'
19 O                                   57 'QTY'
20 O                                   74 'DESCRIPTION'
   O        D  1  13 50
   O                                   41 'ORDERED'
   O                                   59 'SHIPPED'
   O        D  1  13
   O                         ITEM  2   30
```

*Number of sheets per pad may vary slightly

BEST COPY AVAILABLE

Assignment Sheet 2: Source Program Solution for Problem 2

IBM International Business Machines Corporation

RPG OUTPUT SPECIFICATIONS

GX21-9090-4 UM/050*
Printed in U.S.A.

Program
Programmer
Date
Keying Instruction: Graphic / Key
Card Electro Number
Page 10 of 11
Program Identification (75 76 77 78 79 80)

Commas	Zero Balances to Print	No Sign	CR	–
Yes	Yes	1	A	J
Yes	No	2	B	K
No	Yes	3	C	L
No	No	4	D	M

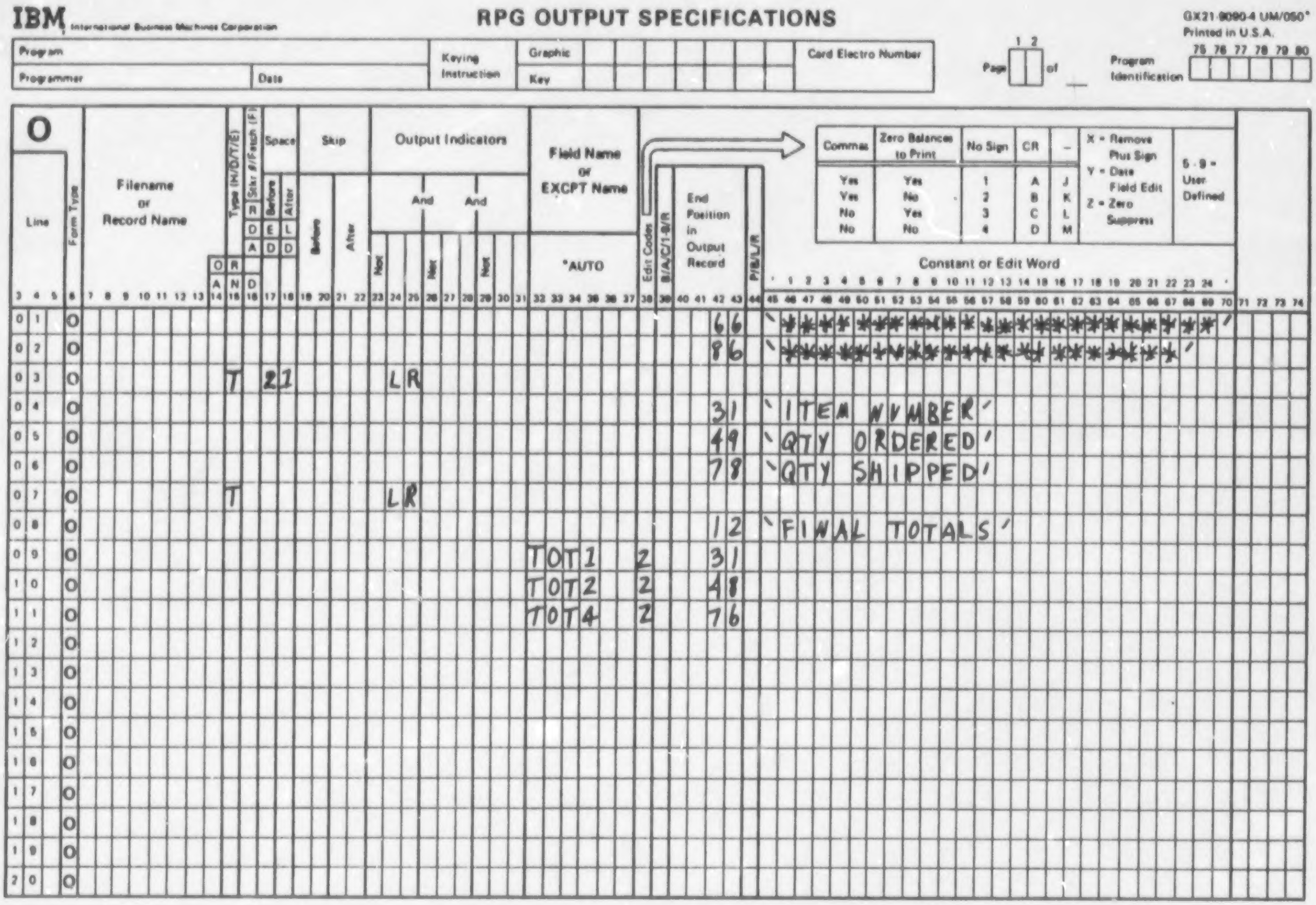
X = Remove Plus Sign
Y = Date Field Edit
Z = Zero Suppress
5-9 = User Defined

Line	Form Type	Filename or Record Name	Type (H/D/T/E)	Space After	Output Indicators	Field Name or EXCPT Name	Edit Codes	Blank After (B/A/C/1-9/R)	End Position in Output Record	Constant or Edit Word
01	O					QTYO	Z		40	
02	O					QTYS	Z		58	
03	O					DESC			86	
04	O				13 70 N32				113	'ITEM NOT ON MASTER FILE'
05	O				13 31				111	'QUANTITY ORDERED ZERO'
06	O				13 32				112	'QUANTITY ORDERED OVER '
07	O				13 32				116	'1000'
08	O		D	1	13 70 32					
09	O				13 70 32				113	'ITEM NOT ON MASTER FILE'
10	O		T	1	L1 N26					
11	O								12	' '
12	O		T	1	L1					
13	O								12	'ORDER TOTALS'
14	O					ITEMT	3	B	30	
15	O					QTYOT	3	B	40	
16	O					QTYST	3	B	58	
17	O								76	'ORDER VALUE'
18	O					QUALT	3	B	86	
19	O				80 N81 N87				114	'CUSTOMER EXCEEDED CREDIT'
20	O				80 N81 N87				124	' LIMIT BY '
	O				80 N81 N87	EXCESS	1	B	132	
	O				39				110	'MISSING ITEM RECORDS'
	O		T	1	L1					
	O								22	'************************'
	O								44	'************************'

*Number of sheets per pad may vary slightly

BEST COPY AVAILABLE

Assignment Sheet 2: Source Program Solution for Problem 2



IBM International Business Machines Corporation

RPG OUTPUT SPECIFICATIONS

GX21-9090-4 UM/050*
Printed in U.S.A.

Line	Form Type	Filename or Record Name	Type (H/D/T/E)	Space Before	Space After	Skip Before	Skip After	Output Indicators	Field Name or EXCPT Name	Edit Codes	End Position in Output Record	Constant or Edit Word
01	O										66	'************************'
02	O										86	'**********************'
03	O		T	2	1			LR				
04	O										31	'ITEM NUMBER'
05	O										49	'QTY ORDERED'
06	O										78	'QTY SHIPPED'
07	O		T					LR				
08	O										12	'FINAL TOTALS'
09	O								TOT1	2	31	
10	O								TOT2	2	48	
11	O								TOT4	2	76	
12	O											
13	O											
14	O											
15	O											
16	O											
17	O											
18	O											
19	O											
20	O											

DATA ENTRY AND EDIT PROGRAMS

UNIT XVII

EXAMINATION

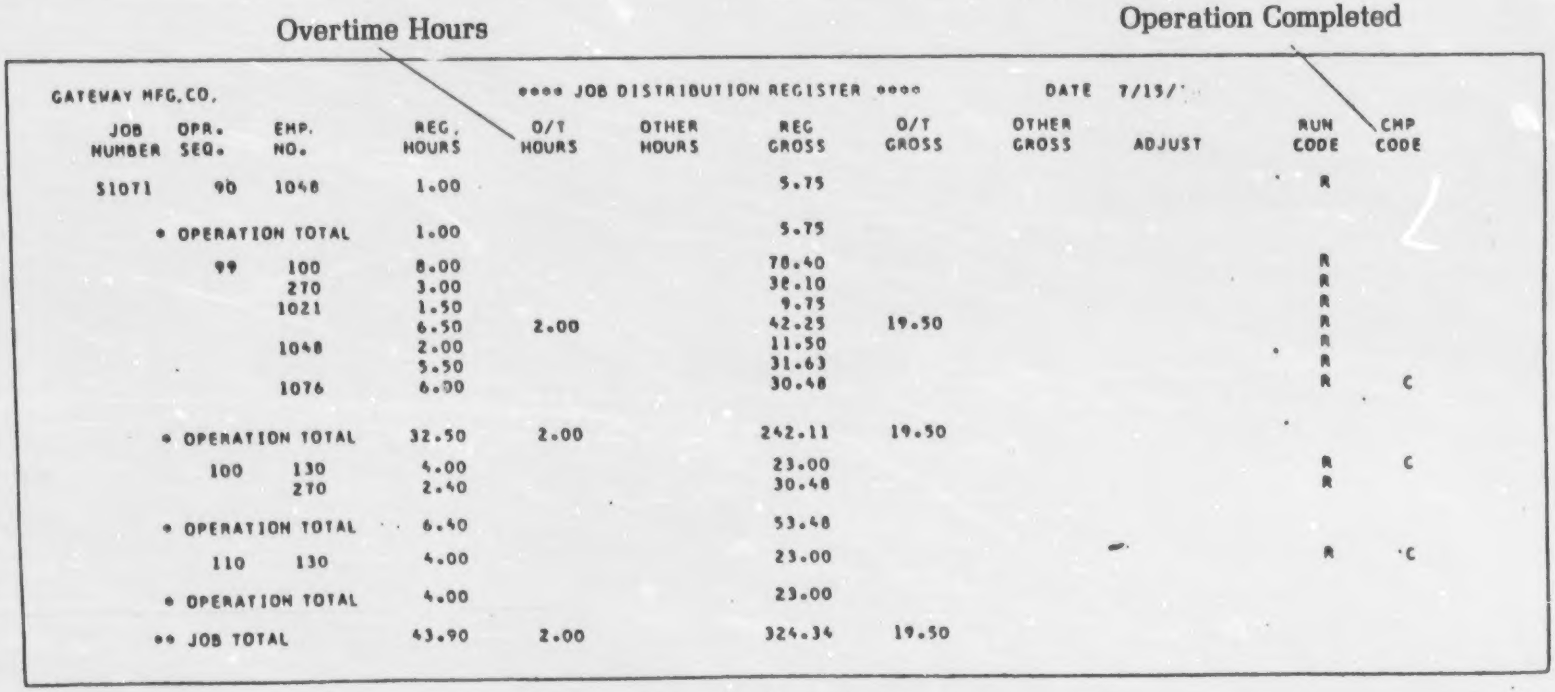
1. Explain the function of data entry.

2. Describe a typical data processing routine that is performed daily in a small business.

3.1 How is edit listing used?

3.2 List 3 types of errors that are detected by edit listing.

Miscellaneous Payroll Reports (cont'd)


Overtime Hours
Operation Completed

```
GATEWAY MFG.CO.                      **** JOB DISTRIBUTION REGISTER ****       DATE  7/13/

  JOB    OPR.    EMP.       REG.     O/T     OTHER      REG      O/T     OTHER                RUN   CMP
NUMBER  SEQ.     NO.       HOURS    HOURS    HOURS     GROSS    GROSS    GROSS     ADJUST     CODE  CODE

 51071      90   1048       1.00                        5.75                                   R

     * OPERATION TOTAL      1.00                        5.75

            99    100       8.00                       78.40                                   R
                  270       3.00                       38.10                                   R
                 1021       1.50                        9.75                                   R
                            6.50     2.00              42.25    19.50                          R
                 1048       2.00                       11.50                                   R
                            5.50                       31.63                                   R
                 1076       6.00                       30.48                                   R     C

     * OPERATION TOTAL     32.50     2.00             242.11    19.50

        100       130       4.00                       23.00                                   R     C
                  270       2.40                       30.48                                   R

     * OPERATION TOTAL      6.40                       53.48

        110       130       4.00                       23.00                                   R     C

     * OPERATION TOTAL      4.00                       23.00

    ** JOB TOTAL           43.90     2.00             324.34    19.50
```

TM 1

Business Applications

B illing

I nventory

C ontrol

A ccounts

R eceivable

S ales

A nalysis

748

Financial Reporting Applications

G eneral

L edger

A ccounts

P ayable

P ay

R oll

GENERAL BUSINESS REPORTS

UNIT XVIII

EXAMINATION

I. Match each general business application with the benefit(s) that it achieves.

BENEFITS

______ a. Maintain centralized up-to-date stock information

______ b. Maintain flexible pricing

______ c. Automatic calculation of finance charge

______ d. Analyze and control company payroll costs

______ e. Reduce errors

______ f. Detect poor credit risks

______ g. Invoice/line item profit

______ h. Track profit performance by customer and item

______ i. Facilitate audit procedures

______ j. Comprehensive accounts breakdown

______ k. Reduce inventory investment

______ l. Comparative analysis

______ m. Reduce bad debt loss

______ n. Reorder on a timely, efficient basis

______ o. Track individual salesman's productivity

______ p. Prepare neat, accurate, and timely statements

______ q. Track performance trends by customer and item

BUSINESS APPLICATIONS

1. Order Entry and Invoicing
2. Accounts Receivable
3. Inventory Control
4. Sales Analysis
5. Payroll
6. General Ledger
7. Accounts Payable

2. Match each business report with the purpose/use for which it is intended.

Purpose/Use

_____ a. Provides information to improve collection of customer accounts and reduces losses from bad debts

_____ b. Sent to delinquent accounts as a reminder that the account is overdue

_____ c. Provides neat, accurate, and timely invoices

d. Used to identify each customer's contributions to sales/profits

_____ e. Used by management to identify those inventory items that are contributing the most significant amount of revenue, profit, or investment

_____ f. Required by law on a quarterly and yearly basis

_____ g. Detects errors prior to invoicing and updating master records

_____ h. Used to identify effectiveness of each salesman in terms of gross sales/profits and sales trends

_____ i. Provides an audit trail of all invoices printed in a given period

_____ j. Periodic report used by management to review the overall status of the inventory

_____ k. Shows gross pay, check number assigned, and all deductions withheld to reach the net pay

_____ l. Sent to customers monthly to assist in collecting money owed

_____ m. Used to identify how each item is contributing to gross sales/profit

Business Report

1. Invoice
2. Invoice/sales register
3. Aged trial balance
4. Customer statement
5. Delinquency notice
6. Stock status report
7. A-B-C analysis report
8. Sales analysis and profit analysis by salesman
9. Sales analysis and profit analysis by item
10. Sales analysis and profit analysis by customer
11. Payroll register
12. Federal reports
13. Invoice edit listing

3. List one advantage for each of the following business reports.

   a. Federal reports

   b. Payroll register

   c. Sales analysis and profit analysis by salesman

   d. A-B-C analysis report

   e. Stock status report

   f. Delinquency notice

   g. Customer statement

   h. Aged trial balance

   i. Invoice/sales register

   j. Invoice edit listing

   k. Invoice

GENERAL BUSINESS REPORTS

UNIT XVIII

ANSWERS TO EXAMINATION

1.
   - a. 3
   - b. 1
   - c. 2
   - d. 5
   - e. 7
   - f. 2
   - g. 1
   - h. 4
   - i. 5
   - j. 6
   - k. 3
   - l. 6
   - m. 2
   - n. 3
   - o. 4
   - p. 2
   - q. 4

2.
   - a. 3
   - b. 5
   - c. 1
   - d. 10
   - e. 7
   - f. 12
   - g. 13
   - h. 8
   - i. 2
   - j. 6
   - k. 11
   - l. 4
   - m. 9

3. (Any one of the following advantages for each of the business reports)

   a. Federal reports -
      Reduce clerical effort
      Available on a timely basis
      Reduce errors

   b. Payroll register
      Reduce clerical effort
      Single source of information

c. Sales analysis and profit analysis by salesman
    Generally not available with a manual system
    Allow management to evaluate salesman performance which could increase sales

d. A-B-C analysis report
    Generally not available in a manual system
    Highlights items with largest dollar investment
    Reduce capital tied up in inventory
    Reduce inventory carrying costs
    Highlights items contributing the most significant amount of revenue and profit

e. Stock status report
    Reduce clerical costs - eliminate cardex system
    Reorder on a timely, efficient basis
    Increase customer goodwill

f. Delinquency notice
    Reduce bad debts
    Improve cash flow
    Available on demand
    No manual effort required

g. Customer statement
    Apply cash payments easier - remittance tear slip returned with payment
    Timely - improve cash flow
    Aging of amount due - psychological reminder which may improve cash flow
    Print special messages
    Reduce manual effort

h. Aged trial balance
    Available on demand
    Reduced clerical effort required
    Variety of formats
    Potential to reduce bad debts - identify delinquent customers

i. Invoice/sales register
    No manual effort required (automatically prepared)
    Provides invoice profit in dollars and percent

j. Edit listing report
    Improves accuracy of invoices

k. Invoice
    Improve accuracy
    Improve cash flow
    Clerical savings
    Track cost versus profit daily for each line item
    Maintain flexible pricing structure to meet profit goals

DATA ENTRY AND EDIT PROGRAMS

UNIT XVII

ANSWERS TO EXAMINATION

1.
    a. Captures information in a medium which can be used repeatedly
    b. Encodes information in a standard format which can be processed by a machine

2.
    a. Event - information is created by an event
    b. Transaction - the event is recorded on a source document so that information can be passed from one process to another.
    c. Batch
        1. All transactions are batched (grouped) into a convenient size for further processing.
        2. Transactions may be transcribed to a special keypunching form.
    d. Keypunch - converts source information into machine readable form (card, disk, or diskette)
    e. Verify - rekeys critical information to assure accuracy
    f. Process - loads machine readable information into a machine for processing
    g. Edit Listing - lists of all information with the last page of the report containing machine generated totals and hash totals used to check against original control tape to assure accuracy

3.1 To detect errors in data entry prior to invoicing and updating master records

3.2 (Any 3 of the following)

a. Is the customer number valid?

b. Is the item number valid?

c. Is the item number missing?

d. Is the valid price overridden?

e. Does the invoice profit percent meet company objectives?

GENERAL BUSINESS REPORTS

UNIT XVIII

OBJECTIVES

After completing this unit, the student should have a basic understanding of the purpose, content, and advantages of general automated business reports.

SPECIFIC OBJECTIVES

After studying this unit, the student should be able to:

1.1. List the overall benefits of the automated order entry and invoicing application.

1.2. State the purpose, content, and advantages of the invoice.

1.3. State the use, content, and advantage of the invoice edit listing report.

1.4. State the purpose, content, and advantages of the invoice/sales register.

2.1. List the overall benefits of the automated accounts receivable application.

2.2. State the use, content, and advantages of the aged trial balance.

2.3. State the use, content, and advantages of the customer statement.

2.4. State the use, content, and advantages of the delinquency notice.

3.1. List the overall benefits of the automated inventory control application.

3.2. State the purpose, content, and advantage of stock status report.

3.3. State the use, content, and advantage of A-B-C analysis report.

4.1. List the overall benefits of the automated sales analysis application.

4.2. State the use, content, and advantages of sales analysis and profit analysis by salesman.

4.3. State the use, content, and advantages of sales analysis and profit analysis by item.

4.4. State the use, content, and advantages of sales analysis and profit analysis by customer.

5.1. List the overall benefits of the automated payroll application.

5.2. State the use, content, and advantage of the payroll register.

5.3. State the use, content, and advantage of required federal reports.

5.4. List 4 additional miscellaneous payroll reports.

6.1. List the overall benefits of the automated general ledger application.

7.1. List the overall benefits of the automated accounts payable application.

GENERAL BUSINESS REPORTS

UNIT XVIII

SUGGESTED ACTIVITIES

I. Instructor
   - A. Provide students with objectives and discuss.
   - B. Provide students with information and assignment sheets and discuss.
   - C. Discuss unit and specific objectives.
   - D. Discuss information sheets.
   - E. Provide handouts.

II. Student
   - A. Read objectives.
   - B. Study information sheets and handouts.

INSTRUCTIONAL MATERIALS

I. Unit content
   - A. Objectives
   - B. Notes to the instructor
   - C. Information sheets
   - D. Handouts
     1. HO 1: Sample Invoice
     2. HO 2: Sample Invoice Edit List
     3. HO 3: Sample invoice/Sales Register
     4. HO 4: Sample Aged Trial Balance Report
     5. HO 5: Sample Customer Statement

        6. HO 6: Sample Delinquency Notice
        7. HO 7: Sample Stock Status Review
        8. HO 8: Sample A-B-C Analysis Report
        9. HO 9: Sales Analysis and Profit Analysis by Salesman
        10. HO 10: Sales Analysis and Profit Analysis by Item
        11. HO 11: Sales Analysis and Profit Analysis by Customer
        12. HO 12: Payroll Register
        13. HO 13: Federal Reports
        14. HO 14: Employee's Wages and Tax Statement
        15. HO 15: Miscellaneous Payroll Reports

    E. Transparency masters

        1. TM 1: Business Applications
        2. TM 2: Financial Reporting Applications

    F. Examination

    G. Answers to examination

II. Reference

    A. Consult local computer branch office to obtain more information on applications and their basic output (i.e., IBM, NCR, Burroughs, Digital Equipment Corp., etc).

GENERAL BUSINESS REPORTS

UNIT XVIII

NOTES TO THE INSTRUCTOR

This final unit in the RPG section may be the most significant for the student because it is designed to give examples and information about the general automated business reports used in daily office operations. These examples lend a touch of reality as to one of the main purposes for computer usage showing that computers can do something besides play games and perform mathematical calculations.

TM 1 and TM 2 make use of acronyms to sum up applications of the business world. TM 1: BICARSA (pronounced Bi car sa) denotes the applications on the business or operation side of a company. TM 2: GLAPPR (pronounced Glap purr) represents applications on the financial side of business.

GENERAL BUSINESS REPORTS

UNIT XVIII

INFORMATION SHEETS

I. Computerized Applications Reports (TM 1)

- A. Order Entry and Invoicing (Billing)
  - 1. Benefits
    - a. Pricing flexibility
    - b. Clerical savings
    - c. Invoice/line item profit
    - d. Improve cash flow
    - e. Customer good will
  - 2. Invoice (HO 1)
    - a. Purpose - provide neat, accurate, and timely information
    - b. Content
      - 1) Sold to/ship to
      - 2) Miscellaneous line
      - 3) Item number/description
      - 4) Unit of measure
      - 5) Quantity
      - 6) Unit price/extended amount
      - 7) Weight
      - 8) Net sales amount
      - 9) Trade discount
      - 10) Miscellaneous charges
      - 11) Taxes
      - 12) Terms discount
      - 13) Amount due
      - 14) Tear slip - item gross profit percent and dollars
    - c. Advantages
      - 1) Improved accuracy
        - a) Pricing
        - b) Discounts
        - c) Extensions
        - d) Taxes

        2) Improve cash flow

            a) Timely invoices
            b) Increase profits

        3) Clerical savings

            a) Enter information once

        4) Track cost versus profit daily for each line item

            a) Increase profits

        5) Maintain flexible pricing structure to meet profit goals

            a) Quantity break discounts
            b) Discount or markup percentages
            c) Contract prices

3. Invoice edit listing (HO 2)

    a. Purpose - to detect errors prior to invoicing and updating master records

    b. Content

        1) Is the customer number valid?
        2) Is the item number valid?
        3) Is the item number missing?
        4) Is valid price overridden?
        5) Does the invoice profit percent meet company objectives?

    c. Advantage - improves accuracy of invoices

4. Invoice/Sales Register (HO 3)

    a. Purpose - provides an audit trail of all invoices (summary) printed in a given period (usually daily)

    b. Content

        1) Invoice number
        2) Customer name/number
        3) P.O. number
        4) Sales amount
        5) Special charges
        6) Freight
        7) Discount
        8) Total taxes
        9) Invoice total
        10) Invoice cost
        11) Invoice profit in dollars and percent

c. Advantages

1) No manual effort required (automatically prepared)
2) Provides invoice profit in dollars and percent

B. Accounts Receivable

1. Benefits

a. Improve cash flow

b. Reduce bad debt losses

c. Detect poor credit risks

d. Save interest on borrowed working capital

e. Maintain customer good will

f. Reduce clerical efforts and improve accuracy

g. Prepare neat, accurate, and timely statements

h. Provide automatic calculation of finance charge

2. Aged Trial Balance (HO 4)

a. Use - provides information to improve collection of customer accounts and reduce losses from bad debts through automatic aging of each customer account

b. Content

1) Customer number
2) Customer name
3) Total due
4) Over 30
5) Over 60
6) Over 90
7) Over 120
8) Totals for each category
9) Previous balance (last month's outstanding A/R)
10) Current charges
11) Current payments
12) Current adjustments
13) Detailed aged trial balance format

a) Age of outstanding invoices
b) Date of last payment
c) Miscellaneous data

c. Advantages

1) Available on demand

2) Reduce clerical effort required
3) Variety of formats
4) Potential to reduce bad debts - identify delinquent customers

3. Customer Statements (HO 5)

a. Use - sent to customers monthly to assist in collecting money owed

b. Content

1) Customer name and account number
2) Current month transactions

a) Invoice number and date
b) Payments

3) Aging

a) Current
b) Over 30, 60, 90, 120

4) Total amount due

5) Remittance attachment - invoice detail (number and amount)

6) Special one time messages - example: "Shop Early for X-Mas," etc.

c. Advantages

1) Apply cash payments easier - remittance tear slip returned with payment

2) Timely - improve cash flow

3) Aging of amount due - psychological reminder which may improve cash flow

4) Print special messages

5) Reduce manual effort

4. Delinquency Notice (HO 6)

a. Use - sent to delinquent accounts as a reminder that the account is overdue

b. Content - amount due may or may not be aged

c. Advantages

1) Reduce bad debts
2) Improve cash flow
3) Available on demand
4) No manual effort required

C. Inventory Control

1. Benefits

   a. Reduce inventory investment

   b. Maintain centralized, up-to-date stock information

   c. Reorder on a timely, efficient basis

   d. Reduce clerical costs

   e. Maintain customer goodwill

2. Stock Status Report (HO 7)

   a. Purpose - periodic report used by management to review the overall status of the inventory

   b. Content

      1) Item number
      2) Vendor number
      3) Description
      4) Weight
      5) Unit of measure
      6) Date of last sale
      7) Item class
      8) On hand
      9) On order
      10) On back order
      11) Reserve
      12) Quantity available
      13) Unit cost
      14) Quantity sold this month
      15) Quantity sold year to date
      16) Exception conditions

         a) On hand zero or negative
         b) On hand below minimum
         c) Cost deviation beyond user specified percentage
         d) Quantity on back order

      17) Printed in various sequences

         a) Item number
         b) Item number within item class
         c) Item number within item class within vendor number

c. Advantages

1) Reduce clerical costs - eliminate cardex system

2) Reorder on a timely, efficient basis

(a) Tool for purchasing
(b) Bulk buying

3) Increase customer goodwill

(a) Answer inquiry on status of item
(b) Not precise because of activity occurring since report was printed

3. A-B-C Analysis Report (HO 8)

a. Use - used by management to identify those inventory items that are contributing the most significant amount of revenue, profit, or investment

b. Content

1) Item No. and description
2) Rank
3) Cumulative % of items
4) Annual units
5) Unit cost
6) Annual usage in $
7) Cumulative usage in $
8) Cumulative usage in %
9) On hand value in $
10) Cumulative value in %

c. Advantages

1) Generally not available in a manual system
2) Highlights items with largest dollar investment
3) Reduce capital tied up in inventory
4) Reduce inventory carrying costs
5) Highlights items contributing the most significant amount of revenue and profit

D. Sales Analysis

1. Benefits

a. Track performance trends by customer and item

b. Track profit performance by customer, salesman, and item

c. Help evaluate sales organization effectiveness, including territory assignments and product line coverage

d. Track individual salesman's productivity

e. Pinpoint problems for timely management attention

2. Sales Analysis and Profit Analysis by Salesman (HO 9)

   a. Use - to identify effectiveness of each salesman in terms of gross sales, gross profit, and sales trends

   b. Content - explain report headings on both reports

   c. Advantages

      1) Generally not available with a manual system

      2) Allow management to evaluate salesman performance, which could increase sales

3. Sales Analysis and Profit Analysis by item (HO 10)

   a. Use - to identify how each item is contributing to gross sales and gross profit

   b. Content - explain report headings on both reports

   c. Advantages

      1) Generally not available with a manual system

      2) Allow management to identify how items and classes of items are contributing to sales and profit (product line coverage)

      3) Show trends - sales/profits up or down as compared to previous year

      4) Effective use of this information will result in increased sales and profits.

4. Sales Analysis and Profit Analysis by Customer (HO 11)

   a. Use - to identify each customer's contribution to sales and gross profit

   b. Content - explain report headings on both reports

- c. Advantages
  - 1) Generally not available with a manual system
  - 2) Allow management to identify sales/profit contributions by customer
  - 3) Should the company offer low profit customers discounts and free delivery service?
  - 4) Do we want this customer's business?
  - 5) Effective use of this information could result in increased sales and profits.

E. Payroll (TM 2)

- 1. Benefits
  - a. Analyze and control company payroll costs
  - b. Produce clerical savings
  - c. Facilitate audit procedures
- 2. Payroll Register (HO 12)
  - a. Use - shows gross pay, check number assigned, and all deductions withheld to reach the net pay. Final control totals are shown for balancing to other reports
  - b. Content
    - 1) Employee number and name
    - 2) Hours worked
    - 3) Gross pay - regular; vacation
    - 4) Taxable adjustment - example: gifts, errors
    - 5) Nontaxable adjustment- example: travel expenses
    - 6) Total gross
    - 7) Taxable gross
    - 8) Federal tax
    - 9) FICA tax - Federal Insurance Contribution Act

768

10) State tax

11) Local tax - one time a year (Pittsburgh $10 per year); percentage of gross

12) Union deduction

13) Miscellaneous deductions - savings bonds, life insurance, hospitalization insurance

14) Net pay

15) Check number

16) Final totals of each column

c. Advantages

1) Reduce clerical effort

2) Single source of information

3. Federal Reports (941A and W-2) (HO 13, 14)

a. Use - required by Federal law on a quarterly and yearly basis

b. Content (W-2)

1) Employee name
2) Social Security number
3) Quarterly wages paid

c. Content (W-2)

1) Federal income tax withheld
2) Yearly wages paid
3) FICA tax withheld
4) Total FICA wages

d. Advantages

1) Reduce clerical effort
2) Available on a timely basis
3) Reduce errors

4. Miscellaneous Payroll Reports (HO 15)

a. Miscellaneous deduction register
b. YTD stub on payroll check
c. Labor distribution register
d. Job distribution register

F. General Ledger Benefits

1. Clerical savings

2. Comparative analysis

   a. Previous year
   b. Budget

3. Comprehensive account breakdown

4. Common sizing

5. Timely financial reports

G. Accounts Payable Benefits

1. Clerical savings

   a. Automatic distribution function

   b. Print check/remittance statements

2. Reduce errors

   a. Take advantage of discounts

   b. Maintain good credit rating

3. Timely management reports

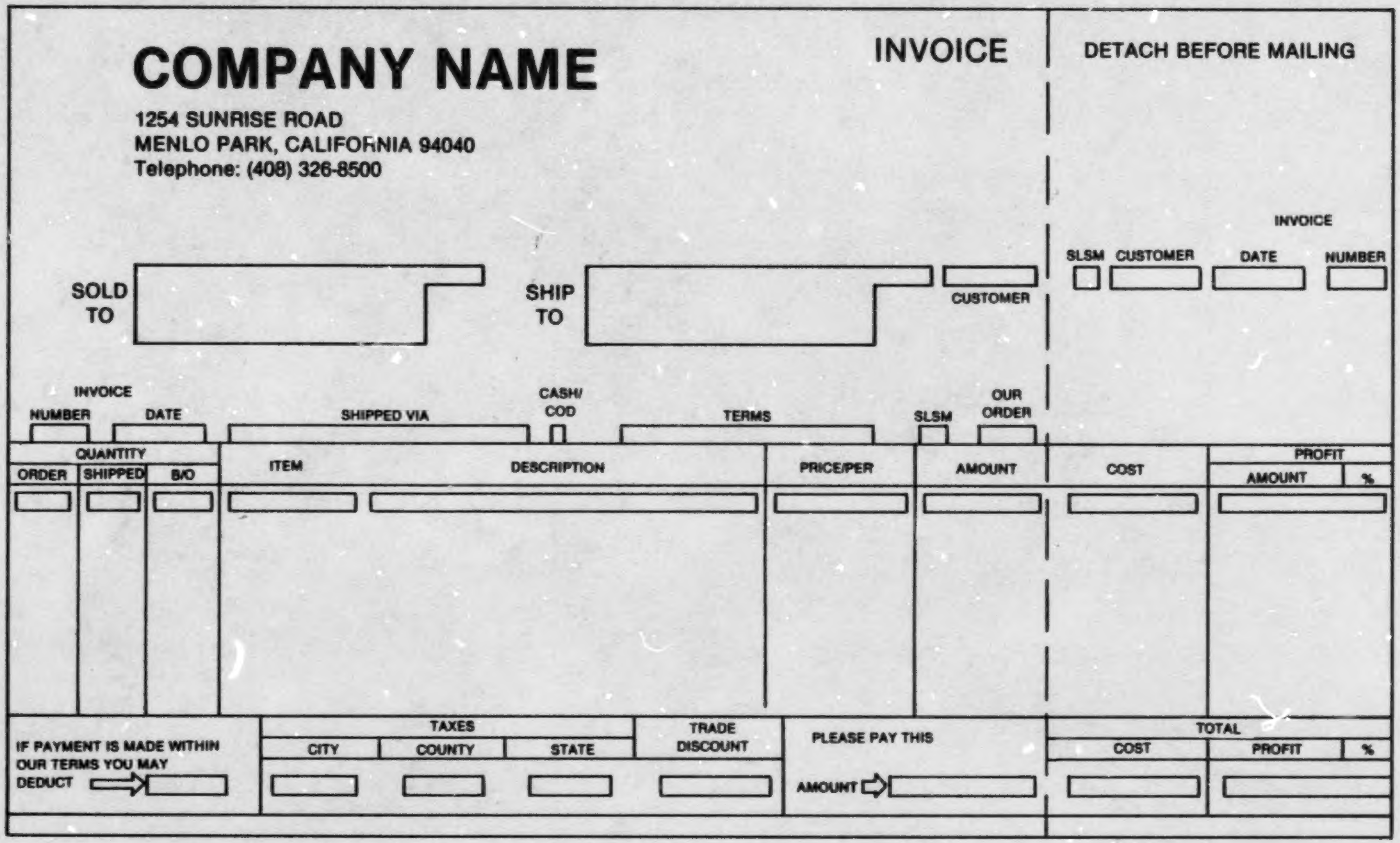
   a. Better control of expenses

   b. Vendor analysis

   c. Cash requirements report (various formats)

HO 1

Sample Invoice



COMPANY NAME

1254 SUNRISE ROAD
MENLO PARK, CALIFORNIA 94040
Telephone: (408) 326-8500

INVOICE

DETACH BEFORE MAILING

SOLD TO

SHIP TO

CUSTOMER

SLSM	CUSTOMER	INVOICE DATE	INVOICE NUMBER

INVOICE NUMBER	INVOICE DATE	SHIPPED VIA	CASH/ COD	TERMS	SLSM	OUR ORDER

QUANTITY ORDER	QUANTITY SHIPPED	QUANTITY B/O	ITEM	DESCRIPTION	PRICE/PER	AMOUNT	COST	PROFIT AMOUNT	PROFIT %

IF PAYMENT IS MADE WITHIN OUR TERMS YOU MAY DEDUCT ⇨

TAXES CITY	TAXES COUNTY	TAXES STATE	TRADE DISCOUNT

PLEASE PAY THIS AMOUNT ⇨

TOTAL COST	TOTAL PROFIT	%

741

HO 2

Sample Invoice Edit List

1-31-**

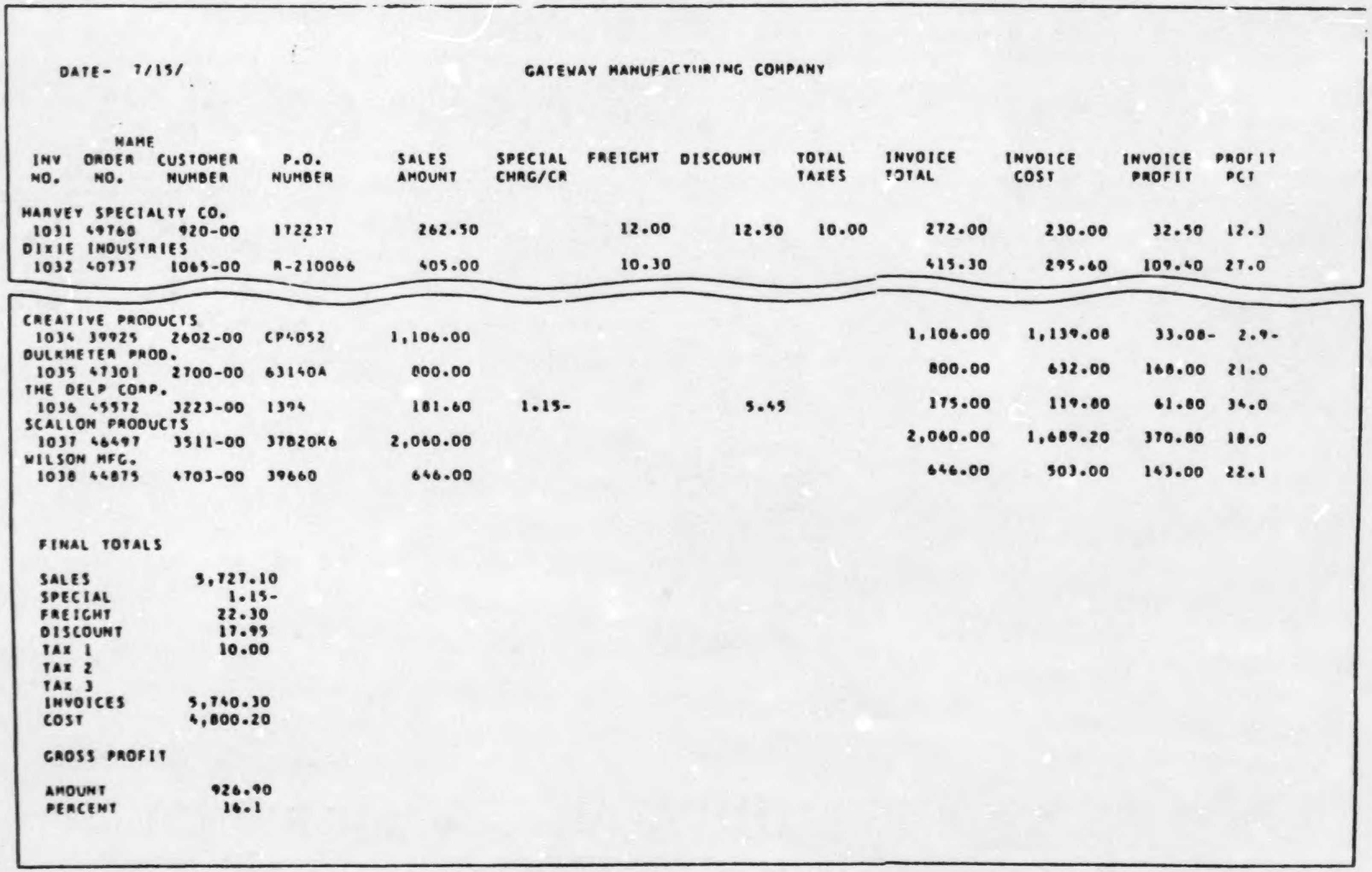
NOTES

Record Type	CUSTOMER NO. ITEM NO.	NAME DESCRIPTION	INVOICE NO. QUANTITY	U/M	ERROR NOTES
C	920-00	HARVEY SPECIALTY	1031		
I	01-40770	OIL FILTER	12	EA	
I	01-40810	OIL SEAL	6	EA	
I	01-40821				***INVALID ITEM NO.
I	01-40840	CARB LINKAGE	3	EA	
C	1065-00				***INVALID CUSTOMER NO.
I	01-40860	SPRING	50	BX	***UNIT-MEASURE
I	01-40870	FUEL PUMP	10	EA	
I	01-40880	PUMP GASKET	100	EA	*** QUANTITY
I	01-40890	BEARING	100	EA	
C	1076-00	WILSON MFG.	1032		
I	01-40770	OIL FILTER	100	EA	
I	01-40780	FUEL PUMP		EA	*** QTY. MISSING

HASH TOTAL 381

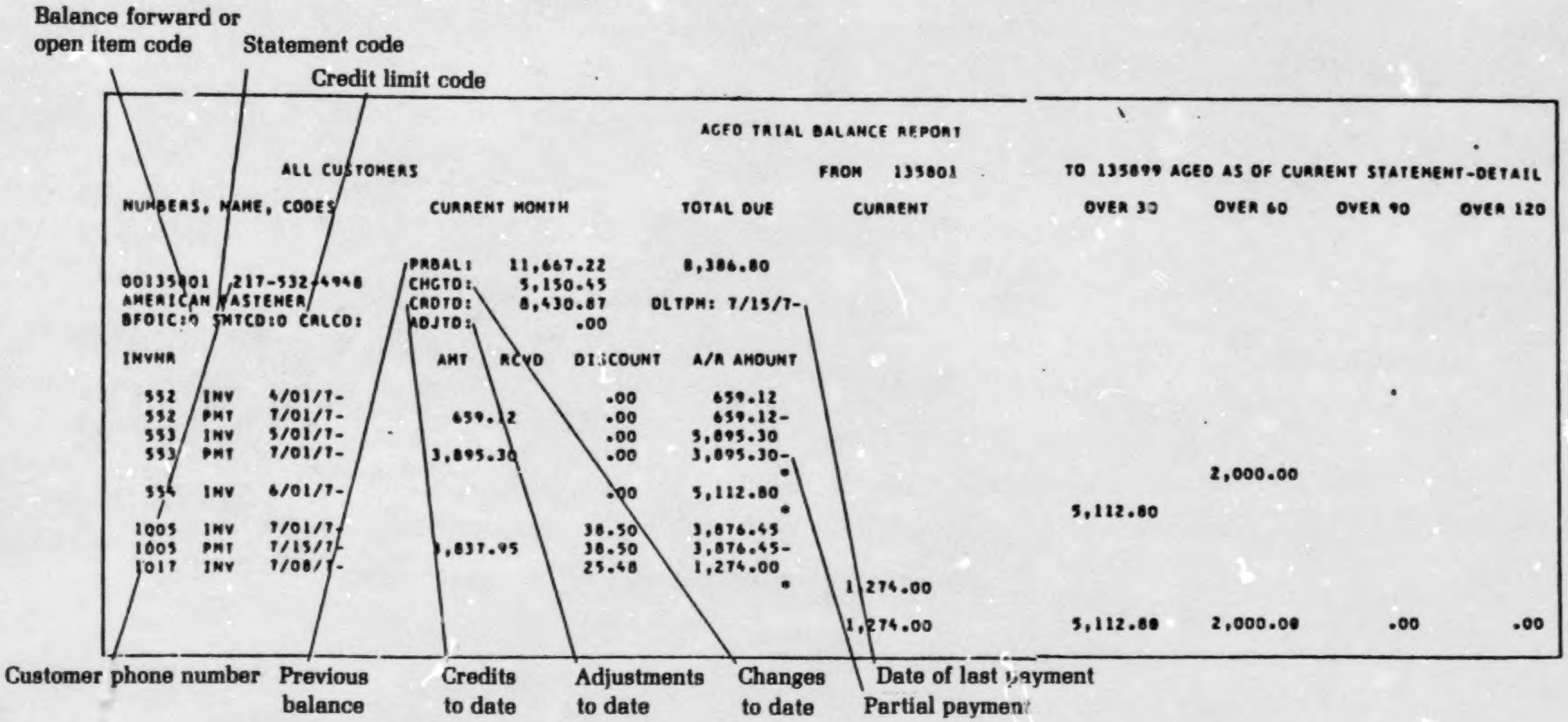
BEST COPY AVAILABLE

Sample Invoice/Sales Register



```
   DATE-  7/15/                          GATEWAY MANUFACTURING COMPANY


          NAME
 INV   ORDER  CUSTOMER    P.O.        SALES     SPECIAL   FREIGHT  DISCOUNT   TOTAL   INVOICE     INVOICE    INVOICE  PROFIT
 NO.   NO.    NUMBER      NUMBER      AMOUNT    CHRG/CR                       TAXES   TOTAL       COST       PROFIT   PCT

HARVEY SPECIALTY CO.
 1031 49760     920-00    172237       262.50              12.00      12.50   10.00     272.00      230.00     32.50  12.3
DIXIE INDUSTRIES
 1032 40737    1065-00    R-210066     405.00              10.30                        415.30      295.60    109.40  27.0

CREATIVE PRODUCTS
 1034 39925    2602-00  CP4052       1,106.00                                         1,106.00    1,139.08     33.08-  2.9-
DULKMETER PROD.
 1035 47301    2700-00  63140A         800.00                                           800.00      632.00    168.00  21.0
THE DELP CORP.
 1036 45572    3223-00  1394           181.60    1.15-                5.45              175.00      119.80     61.80  34.0
SCALLON PRODUCTS
 1037 46497    3511-00  37820K6      2,060.00                                         2,060.00    1,689.20    370.80  18.0
WILSON MFG.
 1038 44875    4703-00  39660          646.00                                           646.00      503.00    143.00  22.1


  FINAL TOTALS

  SALES          5,727.10
  SPECIAL           1.15-
  FREIGHT          22.30
  DISCOUNT         17.95
  TAX 1            10.00
  TAX 2
  TAX 3
  INVOICES       5,740.30
  COST           4,800.20

  GROSS PROFIT

  AMOUNT          926.90
  PERCENT          16.1
```

Sample Aged Trial Balance Report


Balance forward or open item code
Statement code
Credit limit code
Customer phone number
Previous balance
Credits to date
Adjustments to date
Changes to date
Date of last payment
Partial payment

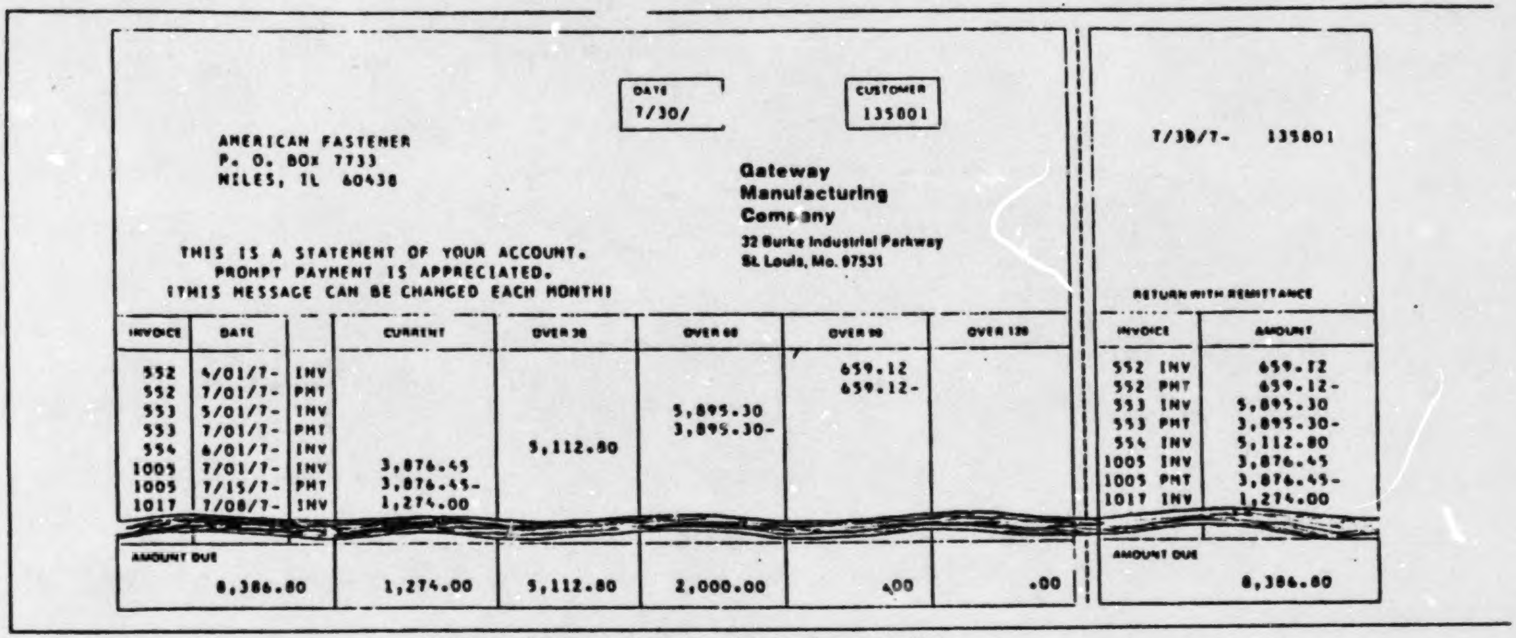
```
                                          AGED TRIAL BALANCE REPORT

               ALL CUSTOMERS                                        FROM    135801         TO 135899 AGED AS OF CURRENT STATEMENT-DETAIL

NUMBERS, NAME, CODES         CURRENT MONTH            TOTAL DUE      CURRENT                 OVER 30     OVER 60     OVER 90     OVER 120

                             PRBAL:    11,667.22       8,386.80
00135801  217-532-4948       CHGTD:     5,150.45
AMERICAN FASTENER            CRDTD:     8,430.87    DLTPM: 7/15/7-
BFOIC:0 SMTCD:0 CALCD:       ADJTD:          .00

INVNR                              AMT   RCVD   DISCOUNT    A/R AMOUNT

  552   INV   4/01/7-                              .00        659.12
  552   PMT   7/01/7-              659.12          .00        659.12-
  553   INV   5/01/7-                              .00      5,895.30
  553   PMT   7/01/7-            3,895.30          .00      3,895.30-
                                                                   *                                  2,000.00
  554   INV   6/01/7-                              .00      5,112.80
                                                                   *                      5,112.80
 1005   INV   7/01/7-                            38.50      3,876.45
 1005   PMT   7/15/7-            3,837.95        38.50      3,876.45-
 1017   INV   7/08/7-                            25.48      1,274.00
                                                                   *    1,274.00

                                                                        1,274.00          5,112.80    2,000.00         .00         .00
```

744

HO 4

BEST COPY AVAILABLE

Sample Customer Statement



DATE	CUSTOMER
7/30/	135801

AMERICAN FASTENER
P. O. BOX 7733
NILES, IL 60438

Gateway
Manufacturing
Company
32 Burke Industrial Parkway
St. Louis, Mo. 97531

THIS IS A STATEMENT OF YOUR ACCOUNT.
PROMPT PAYMENT IS APPRECIATED.
(THIS MESSAGE CAN BE CHANGED EACH MONTH)

INVOICE	DATE		CURRENT	OVER 30	OVER 60	OVER 90	OVER 120
552	4/01/7-	INV				659.12	
552	7/01/7-	PMT				659.12-	
553	5/01/7-	INV			5,895.30		
553	7/01/7-	PMT			3,895.30-		
554	6/01/7-	INV		5,112.80			
1005	7/01/7-	INV	3,876.45				
1005	7/15/7-	PMT	3,876.45-				
1017	7/08/7-	INV	1,274.00				
AMOUNT DUE 8,386.80			1,274.00	5,112.80	2,000.00	.00	.00

7/30/7- 135801

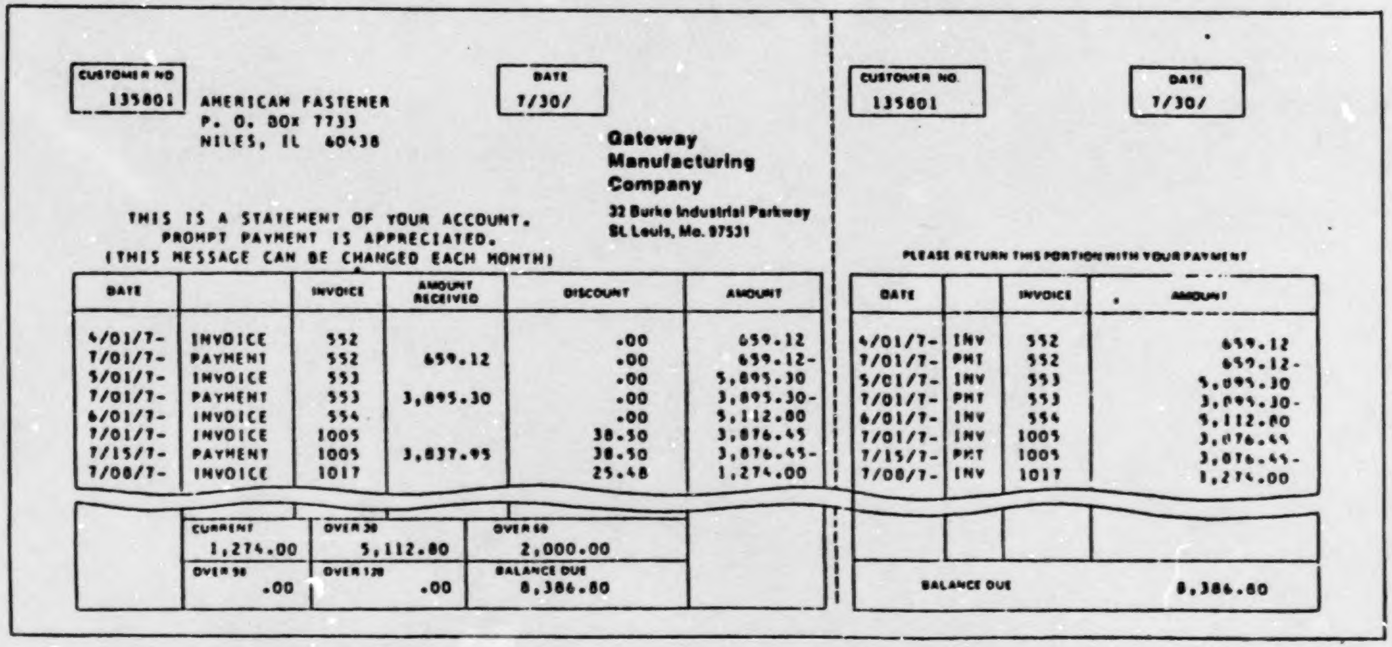
RETURN WITH REMITTANCE

INVOICE	AMOUNT
552 INV	659.12
552 PMT	659.12-
553 INV	5,895.30
553 PMT	3,895.30-
554 INV	5,112.80
1005 INV	3,876.45
1005 PMT	3,876.45-
1017 INV	1,274.00
AMOUNT DUE 8,386.80	

Your Customer Statements can be printed in either of two formats, as you elect.

BEST COPY AVAILABLE

Sample Invoice/Sales Register (cont'd)



CUSTOMER NO
135801

AMERICAN FASTENER
P. O. BOX 7733
NILES, IL 60438

DATE
7/30/

Gateway Manufacturing Company
32 Burke Industrial Parkway
St. Louis, Mo. 97531

THIS IS A STATEMENT OF YOUR ACCOUNT.
PROMPT PAYMENT IS APPRECIATED.
(THIS MESSAGE CAN BE CHANGED EACH MONTH)

DATE		INVOICE	AMOUNT RECEIVED	DISCOUNT	AMOUNT
4/01/7-	INVOICE	552		.00	659.12
7/01/7-	PAYMENT	552	659.12	.00	659.12-
5/01/7-	INVOICE	553		.00	5,895.30
7/01/7-	PAYMENT	553	3,895.30	.00	3,895.30-
6/01/7-	INVOICE	554		.00	5,112.80
7/01/7-	INVOICE	1005		38.50	3,876.45
7/15/7-	PAYMENT	1005	3,837.95	38.50	3,876.45-
7/08/7-	INVOICE	1017		25.48	1,274.00

CURRENT	OVER 30	OVER 60
1,274.00	5,112.80	2,000.00
OVER 90	OVER 120	BALANCE DUE
.00	.00	8,386.80

CUSTOMER NO.
135801

DATE
7/30/

PLEASE RETURN THIS PORTION WITH YOUR PAYMENT

DATE		INVOICE	AMOUNT
4/01/7-	INV	552	659.12
7/01/7-	PMT	552	659.12-
5/01/7-	INV	553	5,895.30
7/01/7-	PMT	553	3,895.30-
6/01/7-	INV	554	5,112.80
7/01/7-	INV	1005	3,876.45
7/15/7-	PMT	1005	3,876.45-
7/08/7-	INV	1017	1,274.00

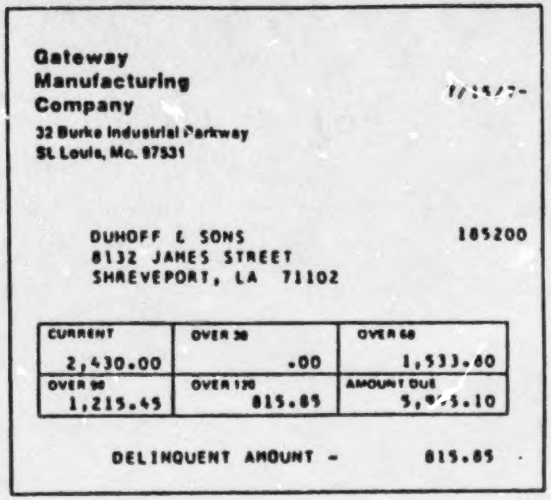
BALANCE DUE 8,386.80

746

BEST COPY AVAILABLE

BEST COPY AVAILABLE

Sample Delinquency Notice



Gateway Manufacturing Company
32 Burke Industrial Parkway
St. Louis, Mo. 97531

7/15/7-

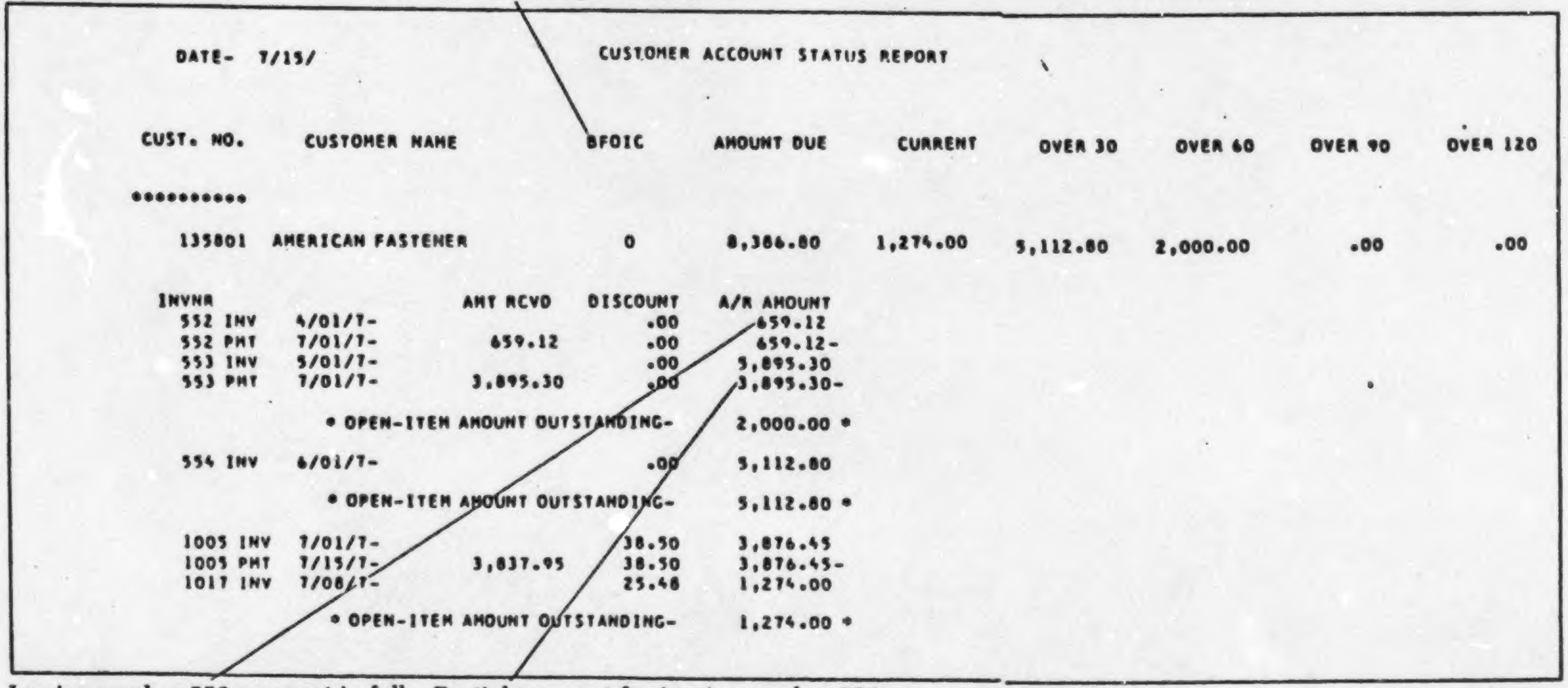
DUHOFF & SONS
8132 JAMES STREET
SHREVEPORT, LA 71102

185200

CURRENT 2,430.00	OVER 30 .00	OVER 60 1,533.60
OVER 90 1,215.45	OVER 120 815.85	AMOUNT DUE 5,9[illegible]5.10

DELINQUENT AMOUNT - 815.85

Balance forward or open item code

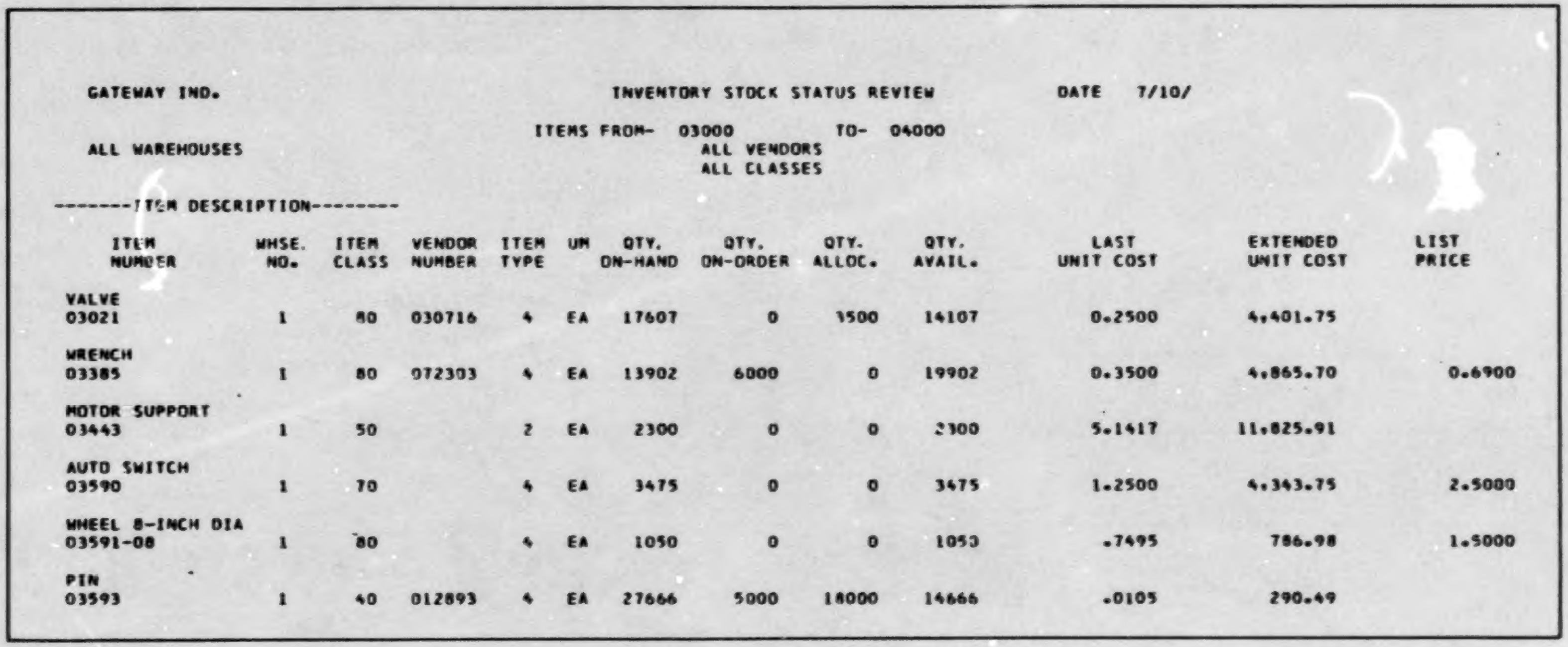


```
  DATE- 7/15/                        CUSTOMER ACCOUNT STATUS REPORT

CUST. NO.   CUSTOMER NAME          BFOIC   AMOUNT DUE    CURRENT    OVER 30    OVER 60    OVER 90   OVER 120

**********

  135801  AMERICAN FASTENER          0      8,386.80    1,274.00   5,112.80   2,000.00      .00        .00

   INVNR                  AMT RCVD   DISCOUNT   A/R AMOUNT
    552 INV   4/01/7-                     .00       659.12
    552 PMT   7/01/7-       659.12        .00       659.12-
    553 INV   5/01/7-                     .00     5,895.30
    553 PMT   7/01/7-     3,895.30        .00     3,895.30-

             * OPEN-ITEM AMOUNT OUTSTANDING-      2,000.00 *

    554 INV   6/01/7-                     .00     5,112.80

             * OPEN-ITEM AMOUNT OUTSTANDING-      5,112.80 *

   1005 INV   7/01/7-                   38.50     3,876.45
   1005 PMT   7/15/7-     3,837.95      38.50     3,876.45-
   1017 INV   7/08/7-                   25.48     1,274.00

             * OPEN-ITEM AMOUNT OUTSTANDING-      1,274.00 *
```

Invoice number 552 payment in full   Partial payment for invoice number 553

747

Sample Stock Status Review



```
  GATEWAY IND.                                      INVENTORY STOCK STATUS REVIEW          DATE   7/10/

                                              ITEMS FROM-  03000          TO-  04000
  ALL WAREHOUSES                                             ALL VENDORS
                                                             ALL CLASSES

-------ITEM DESCRIPTION--------

      ITEM           WHSE.  ITEM   VENDOR  ITEM  UM   QTY.     QTY.      QTY.     QTY.           LAST            EXTENDED        LIST
     NUMBER           NO.   CLASS  NUMBER  TYPE      ON-HAND  ON-ORDER  ALLOC.   AVAIL.         UNIT COST        UNIT COST       PRICE

 VALVE
 03021                 1      80   030716   4    EA   17607        0      3500    14107           0.2500          4,401.75

 WRENCH
 03385                 1      80   072303   4    EA   13902     6000         0    19902           0.3500          4,865.70            0.6900

 MOTOR SUPPORT
 03443                 1      50            2    EA    2300        0         0     2300           5.1417         11,825.91

 AUTO SWITCH
 03590                 1      70            4    EA    3475        0         0     3475           1.2500          4,343.75            2.5000

 WHEEL 8-INCH DIA
 03591-08              1      80            4    EA    1050        0         0     1050            .7495            786.98            1.5000

 PIN
 03593                 1      40   012893   4    EA   27666     5000     18000    14666            .0105            290.49
```

748

BEST COPY AVAILABLE

HO 7

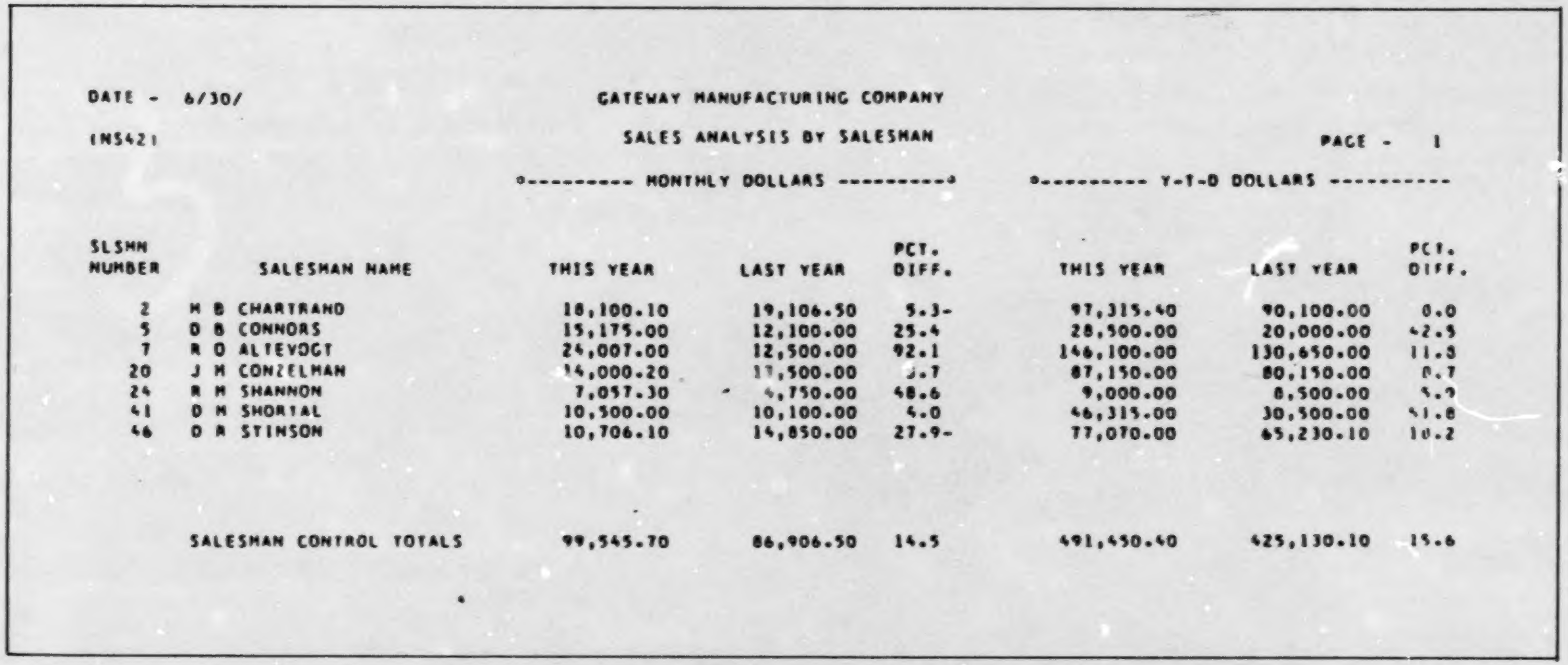
HO 8

Sample A-B-C Analysis Report

```
                                        A-B-C ANALYSIS REPORT


WHSE. ITEM    ITEM        ITEM            ITEM   CUM %  ANNUAL      UNIT          ANNUAL  CUMULATIVE  CUM %   ON HAND   CUM %
 NO. NUMBER   TYPE    DESCRIPTION         COUNT  ITEMS  UNITS       COST          USAGE $   USAGE $   USAGE $ VALUE $   VALUE

                                           20  100.0                                      395,192   100.0     82,891  100.0

  2  4617      3  OIL RED POWDER            1     5.0   20,000     3.850          77,000     77,000   19.5     25,903   31.2
  2  437R      1  OIL SEAL-AMC              2    10.0   35,000     1.539          53,865    130,865   33.1      4,911   37.2
  1  437R      1  OIL SEAL-AMC              3    15.0   35,000     1.537          53,795    184,660   46.7      5,607   44.0
  1  4064      3  CHEMICAL # 40             4    20.0    2,500    15.000          37,500    222,160   56.2     16,125   63.5
  2  4064      3  CHEMICAL # 40             5    25.0    2,500    15.000          37,500    259,660   65.7     14,700   81.2
  1  4007      4  MASTER COMPOUND #7        6    30.0    2,500     9.707          24,268    283,928   71.8      4,360   86.5
  1  49732     1  SYNTHETIC FLANGE          7    35.0   35,000      .524          18,340    302,260   76.5        555   87.2
  2  4030      3  STEEL                     8    40.0  130,000      .138          17,940    320,208   81.0        767   88.1
  2  4725      3  CHEMICAL # DC-5461        9    45.0    3,500     4.799          16,797    337,005   85.3      3,527   92.3
  1  4933H     1  METAL FLANGE             10    50.0   60,000      .221          13,260    350,265   88.6        151   92.5
  2  49503     1  LINE METAL FLANGE        11    55.0   45,000      .260          11,700    361,965   91.6        201   92.8
  2  4TR       3  PAINT                    12    60.0    2,500     4.300          10,750    372,715   94.3      1,849   95.0
  1  49503     1  LINE METAL FLANGE        13    65.0   40,000      .260          10,400    383,115   96.9        259   95.3
  1  4725      3  CHEMICAL # DC-5461       14    70.0    2,000     4.799           9,598    392,713   99.4      3,119   99.1
  1  4016      3  WIRE                     15    75.0    2,000      .348             696    393,409   99.5         96   99.2
  1  4955G     4  GASKET                   16    80.0   48,000      .012             576    393,985   99.7         10   99.2
  1  4V7       1  COMPOUND                 17    85.0    2,500      .172             430    394,415   99.8        136   99.4
  2  4V7       1  COMPOUND                 18    90.0    2,000      .171             342    394,757   99.9        120   99.5
  2  4145C     3  NAPTHA                   19    95.0    2,100      .136             286    395,043  100.0        256   99.8
  1  4145C     3  NAPTHA                   20   100.0    1,200      .124             149    395,192  100.0        149  100.0


                      TOTAL                20                                              395,192 .           82,891
```

BEST COPY AVAILABLE

Sample Analysis and Profit Analysis by Salesman



```
DATE -  6/30/                                GATEWAY MANUFACTURING COMPANY

(NS421)                                      SALES ANALYSIS BY SALESMAN                                     PAGE -    1

                                       *---------- MONTHLY DOLLARS ----------*      *---------- Y-T-D DOLLARS ----------

SLSMN                                                                   PCT.                                        PCT.
NUMBER        SALESMAN NAME             THIS YEAR        LAST YEAR     DIFF.       THIS YEAR        LAST YEAR     DIFF.

    2    M B CHARTRAND                  18,100.10        19,106.50      5.3-       97,315.40        90,100.00      8.0
    5    D B CONNORS                    15,175.00        12,100.00     25.4        28,500.00        20,000.00     42.5
    7    R O ALTEVOGT                   24,007.00        12,500.00     92.1       146,100.00       130,650.00     11.8
   20    J M CONZELMAN                  14,000.20        13,500.00      3.7        87,150.00        80,150.00      8.7
   24    R M SHANNON                     7,057.30         4,750.00     48.6         9,000.00         8,500.00      5.9
   41    D M SHORTAL                    10,500.00        10,100.00      4.0        46,315.00        30,500.00     51.8
   46    D R STINSON                    10,706.10        14,850.00     27.9-       77,070.00        65,230.10     18.2


         SALESMAN CONTROL TOTALS        99,545.70        86,906.50     14.5       491,450.40       425,130.10     15.6
```

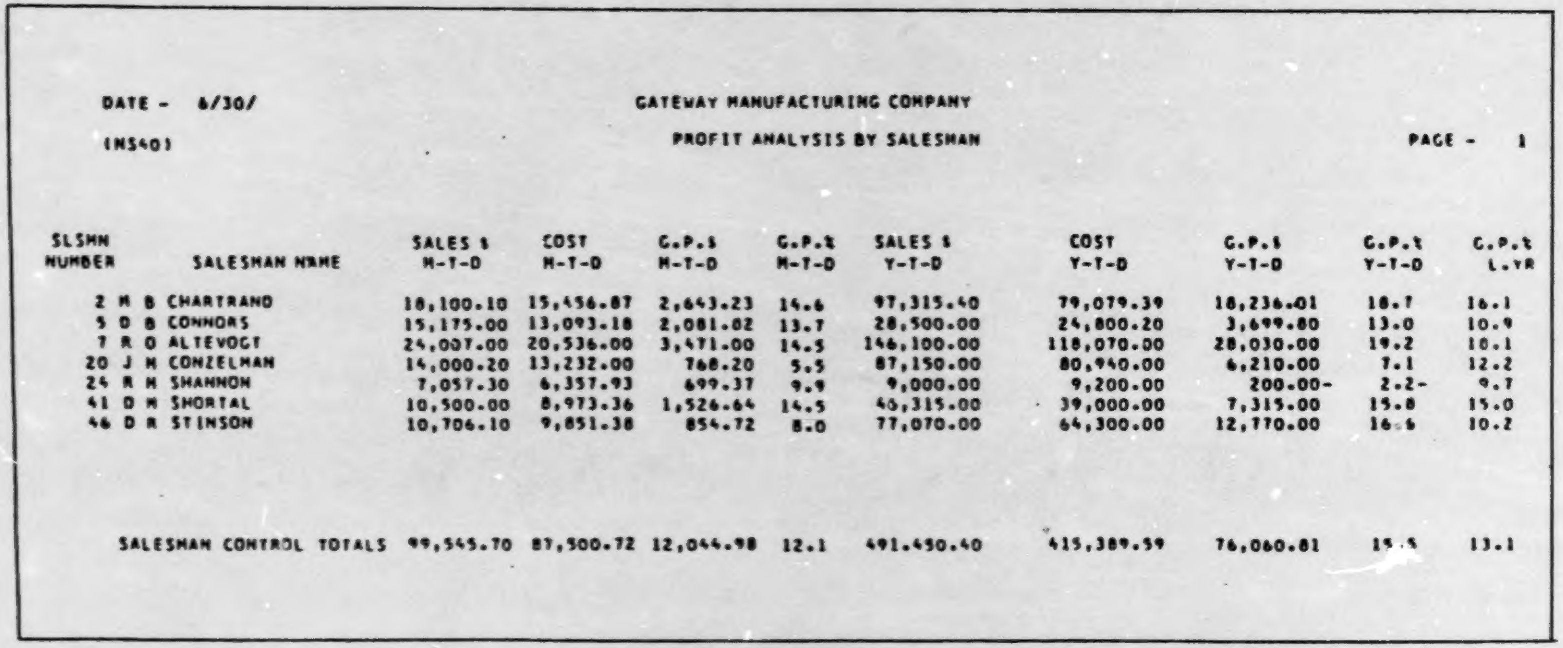
750

HO 9

BEST COPY AVAILABLE

HO 9 (cont'd)

Sample Analysis and Profit Analysis by Salesman (cont'd)



```
   DATE -   6/30/                          GATEWAY MANUFACTURING COMPANY

   (NS40)                                   PROFIT ANALYSIS BY SALESMAN                                        PAGE -   1


 SLSMN                          SALES $    COST       G.P.$      G.P.%    SALES $          COST          G.P.$       G.P.%     G.P.%
 NUMBER      SALESMAN NAME      M-T-D      M-T-D      M-T-D      M-T-D     Y-T-D           Y-T-D         Y-T-D       Y-T-D      L.YR

     2 M B CHARTRAND          18,100.10  15,456.87   2,643.23   14.6     97,315.40       79,079.39     18,236.01     18.7      16.1
     5 D B CONNORS            15,175.00  13,093.18   2,081.02   13.7     28,500.00       24,800.20      3,699.80     13.0      10.9
     7 R O ALTEVOGT           24,007.00  20,536.00   3,471.00   14.5    146,100.00      118,070.00     28,030.00     19.2      10.1
    20 J M CONZELMAN          14,000.20  13,232.00     768.20    5.5     87,150.00       80,940.00      6,210.00      7.1      12.2
    24 R M SHANNON             7,057.30   6,357.93     699.37    9.9      9,000.00        9,200.00        200.00-     2.2-      9.7
    41 D M SHORTAL            10,500.00   8,973.36   1,526.64   14.5     46,315.00       39,000.00      7,315.00     15.8      15.0
    46 D R STINSON            10,706.10   9,851.38     854.72    8.0     77,070.00       64,300.00     12,770.00     16.6      10.2



    SALESMAN CONTROL TOTALS   99,545.70  87,500.72  12,044.98   12.1    491.450.40      415,389.59     76,060.81     15.5      13.1
```

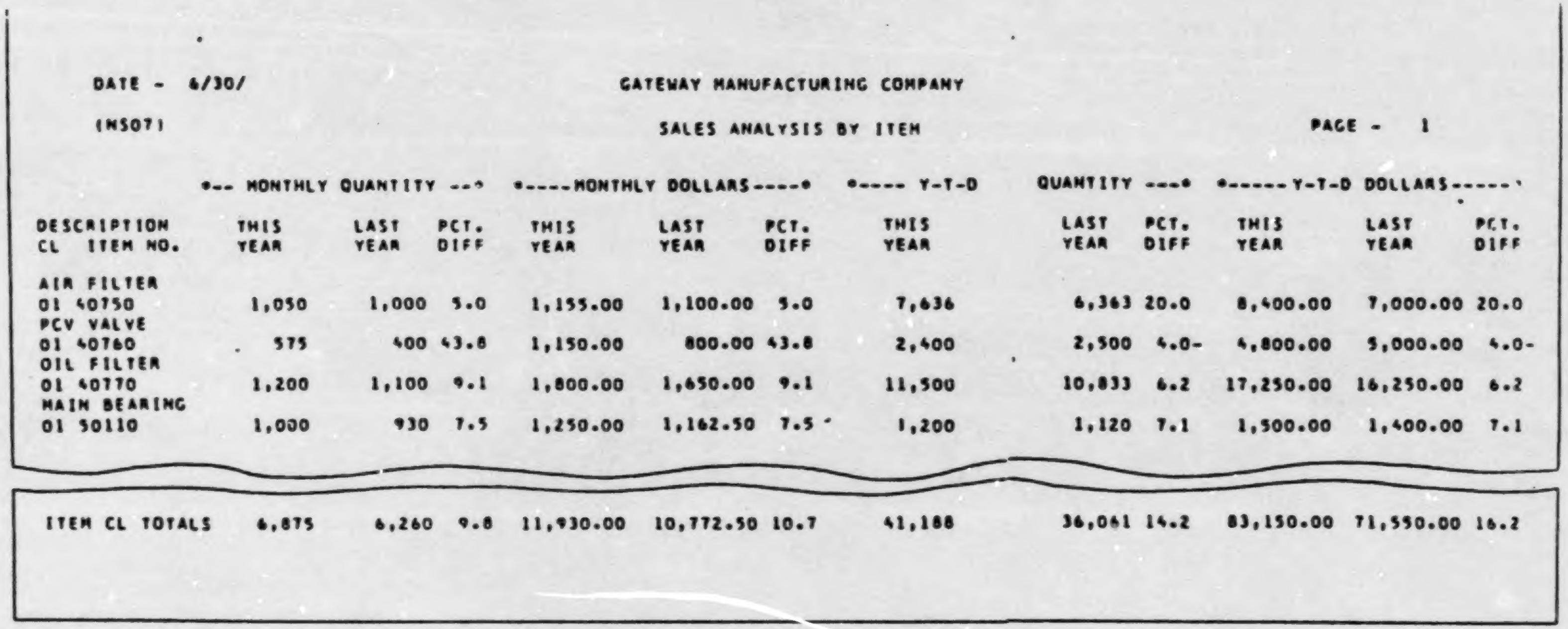
751

BEST COPY AVAILABLE

HO 10

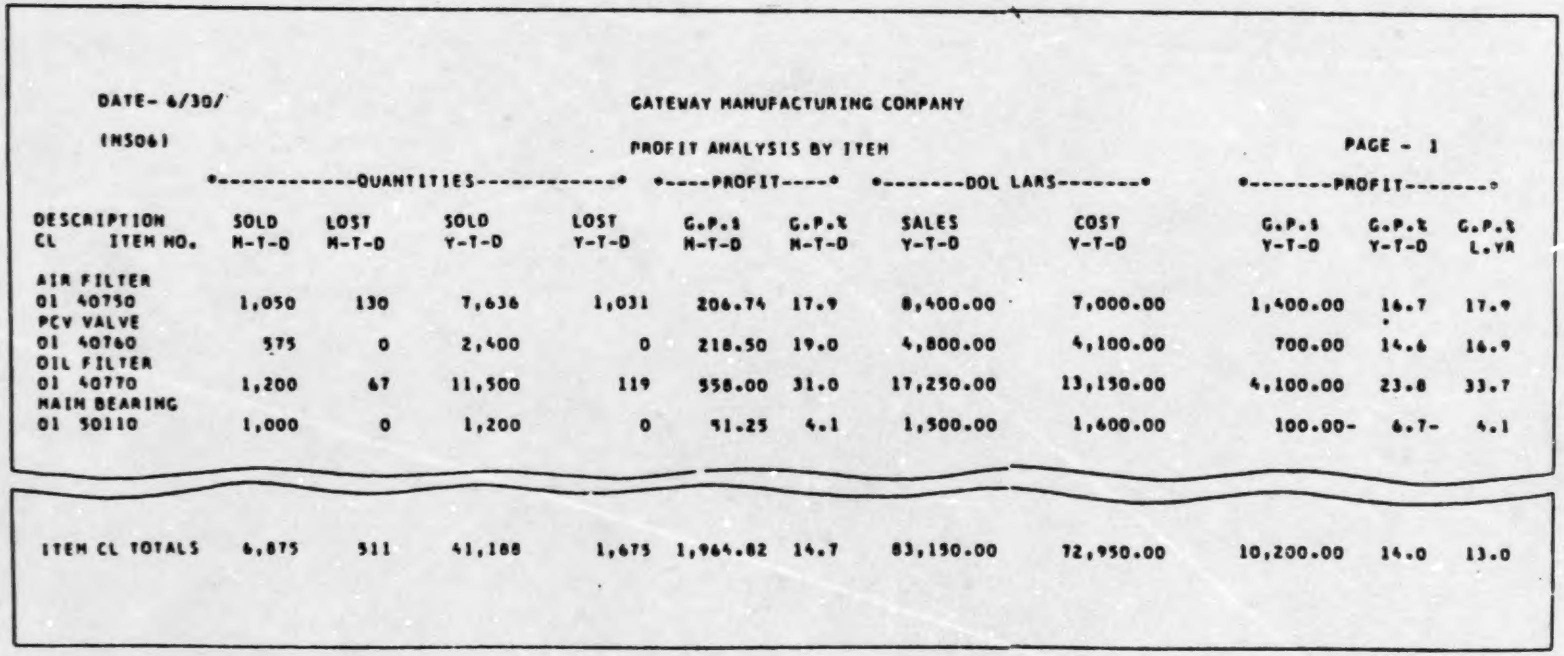
BEST COPY AVAILABLE

Sales Analysis and Profit Analysis by Item



```
DATE -  6/30/                                GATEWAY MANUFACTURING COMPANY

(NS07)                                          SALES ANALYSIS BY ITEM                                   PAGE -   1

               *-- MONTHLY QUANTITY --*  *----MONTHLY DOLLARS----*   *---- Y-T-D    QUANTITY ---*  *-----Y-T-D DOLLARS-----*

DESCRIPTION       THIS      LAST   PCT.    THIS        LAST     PCT.      THIS             LAST   PCT.    THIS        LAST      PCT.
CL  ITEM NO.      YEAR      YEAR   DIFF    YEAR        YEAR     DIFF      YEAR             YEAR   DIFF    YEAR        YEAR      DIFF

AIR FILTER
01 40750         1,050     1,000   5.0   1,155.00   1,100.00   5.0       7,636            6,363 20.0    8,400.00    7,000.00  20.0
PCV VALVE
01 40760           575       400  43.8   1,150.00     800.00  43.8       2,400            2,500  4.0-   4,800.00    5,000.00   4.0-
OIL FILTER
01 40770         1,200     1,100   9.1   1,800.00   1,650.00   9.1      11,500           10,833  6.2   17,250.00   16,250.00   6.2
MAIN BEARING
01 50110         1,000       930   7.5   1,250.00   1,162.50   7.5       1,200            1,120  7.1    1,500.00    1,400.00   7.1

ITEM CL TOTALS   6,875     6,260   9.8  11,930.00  10,772.50  10.7      41,188           36,061 14.2   83,150.00   71,550.00  16.2
```

Sales Analysis and Profit Analysis by Item (cont'd)

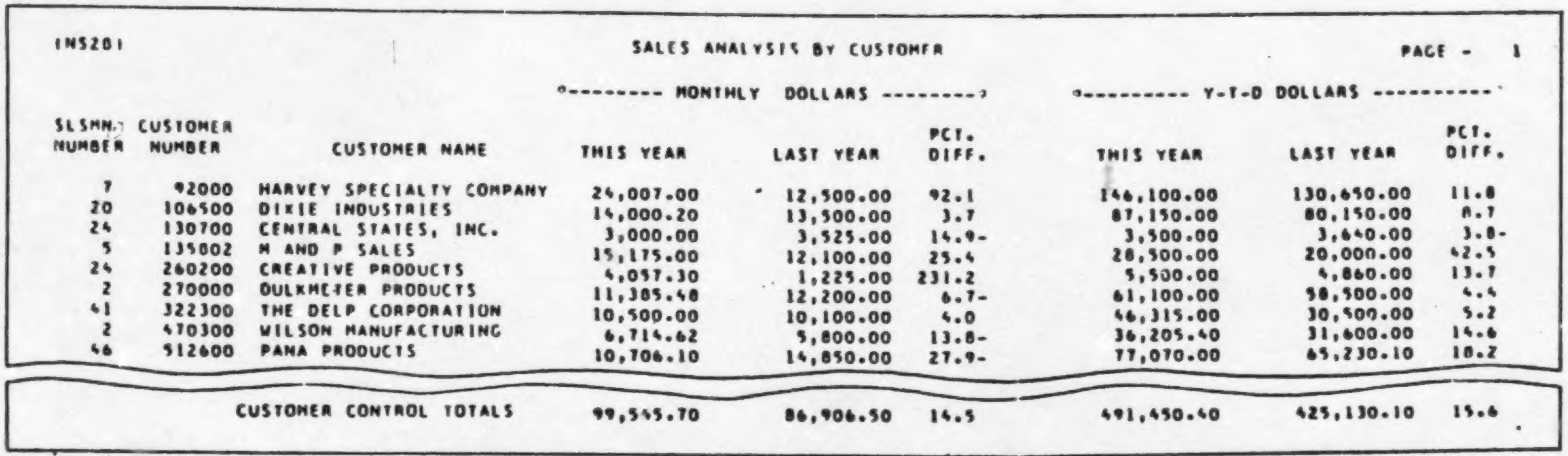


```
DATE- 6/30/                                GATEWAY MANUFACTURING COMPANY

(NS06)                                     PROFIT ANALYSIS BY ITEM                                  PAGE - 1

           *-------------QUANTITIES-------------*  *----PROFIT----*  *-------DOL LARS-------*      *-------PROFIT-------*

DESCRIPTION        SOLD     LOST       SOLD         LOST      G.P.$    G.P.%      SALES           COST          G.P.$     G.P.%   G.P.%
CL    ITEM NO.     M-T-D    M-T-D      Y-T-D        Y-T-D     M-T-D    M-T-D      Y-T-D           Y-T-D         Y-T-D     Y-T-D    L.YR

AIR FILTER
01  40750          1,050      130      7,636        1,031    206.74    17.9      8,400.00       7,000.00      1,400.00    16.7    17.9
PCV VALVE
01  40760            575        0      2,400            0    218.50    19.0      4,800.00       4,100.00        700.00    14.6    16.9
OIL FILTER
01  40770          1,200       67     11,500          119    558.00    31.0     17,250.00      13,150.00      4,100.00    23.8    33.7
MAIN BEARING
01  50110          1,000        0      1,200            0     91.25     4.1      1,500.00       1,600.00        100.00-    6.7-    4.1


ITEM CL TOTALS     6,875      511     41,188        1,675  1,964.82    14.7     83,150.00      72,950.00     10,200.00    14.0    13.0
```

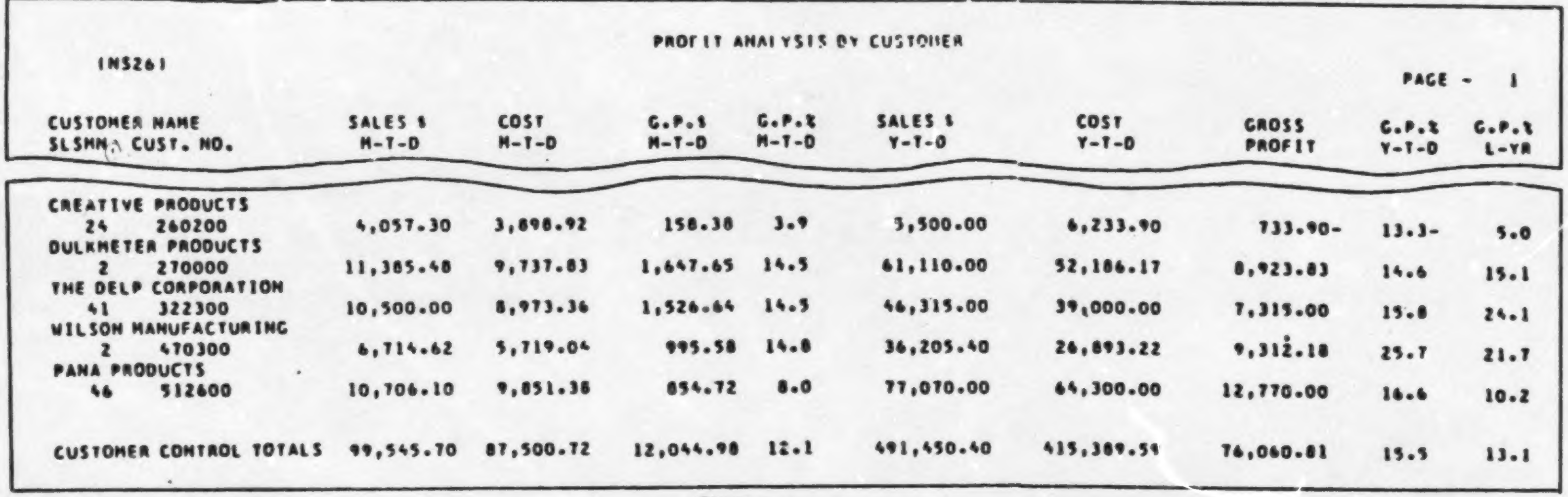
753

BEST COPY AVAILABLE

Sales Analysis and Profit Analysis by Customer



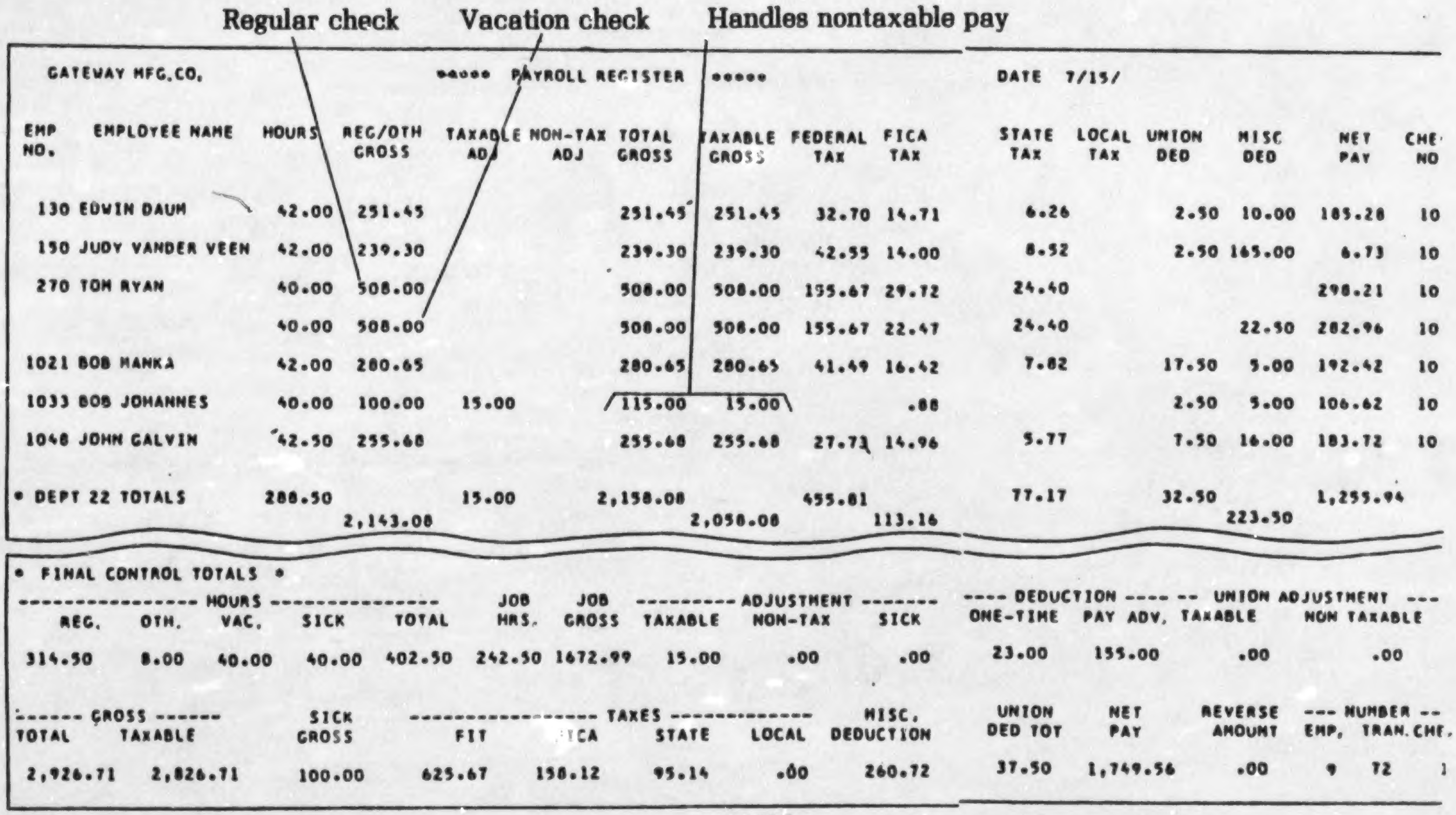
```
INS201                                    SALES ANALYSIS BY CUSTOMER                                          PAGE -   1

                                     *-------- MONTHLY  DOLLARS --------*        *---------- Y-T-D DOLLARS ----------*

SLSMN. CUSTOMER                                                        PCT.                                          PCT.
NUMBER  NUMBER       CUSTOMER NAME           THIS YEAR     LAST YEAR   DIFF.        THIS YEAR      LAST YEAR        DIFF.

    7     92000  HARVEY SPECIALTY COMPANY    24,007.00     12,500.00   92.1        146,100.00     130,650.00        11.8
   20    106500  DIXIE INDUSTRIES            14,000.20     13,500.00    3.7         87,150.00      80,150.00         8.7
   24    130700  CENTRAL STATES, INC.         3,000.00      3,525.00   14.9-         3,500.00       3,640.00         3.8-
    5    135002  M AND P SALES               15,175.00     12,100.00   25.4         28,500.00      20,000.00        42.5
   24    260200  CREATIVE PRODUCTS            4,057.30      1,225.00  231.2          5,500.00       4,860.00        13.7
    2    270000  DULKMETER PRODUCTS          11,385.48     12,200.00    6.7-        61,100.00      58,500.00         4.4
   41    322300  THE DELP CORPORATION        10,500.00     10,100.00    4.0         46,315.00      30,500.00         5.2
    2    470300  WILSON MANUFACTURING         6,714.62      5,800.00   13.8-        36,205.40      31,600.00        14.6
   46    512600  PANA PRODUCTS               10,706.10     14,850.00   27.9-        77,070.00      65,230.10        18.2

              CUSTOMER CONTROL TOTALS        99,545.70     86,906.50   14.5        491,450.40     425,130.10        15.6
```



```
                                             PROFIT ANALYSIS BY CUSTOMER
   INS261
                                                                                                        PAGE -   1

CUSTOMER NAME            SALES $      COST         G.P.$     G.P.%      SALES $        COST            GROSS        G.P.%    G.P.%
SLSMN. CUST. NO.          M-T-D       M-T-D        M-T-D     M-T-D       Y-T-D         Y-T-D           PROFIT       Y-T-D    L-YR

CREATIVE PRODUCTS
   24    260200          4,057.30    3,898.92      158.38     3.9       5,500.00       6,233.90        733.90-     13.3-      5.0
DULKMETER PRODUCTS
    2    270000         11,385.48    9,737.83    1,647.65    14.5      61,110.00      52,186.17       8,923.83     14.6      15.1
THE DELP CORPORATION
   41    322300         10,500.00    8,973.36    1,526.64    14.5      46,315.00      39,000.00       7,315.00     15.8      24.1
WILSON MANUFACTURING
    2    470300          6,714.62    5,719.04      995.58    14.8      36,205.40      26,893.22       9,312.18     25.7      21.7
PANA PRODUCTS
   46    512600         10,706.10    9,851.38      854.72     8.0      77,070.00      64,300.00      12,770.00     16.6      10.2

CUSTOMER CONTROL TOTALS 99,545.70   87,500.72   12,044.98    12.1     491,450.40     415,389.59      76,060.81     15.5      13.1
```

BEST COPY AVAILABLE

Payroll Register



Regular check

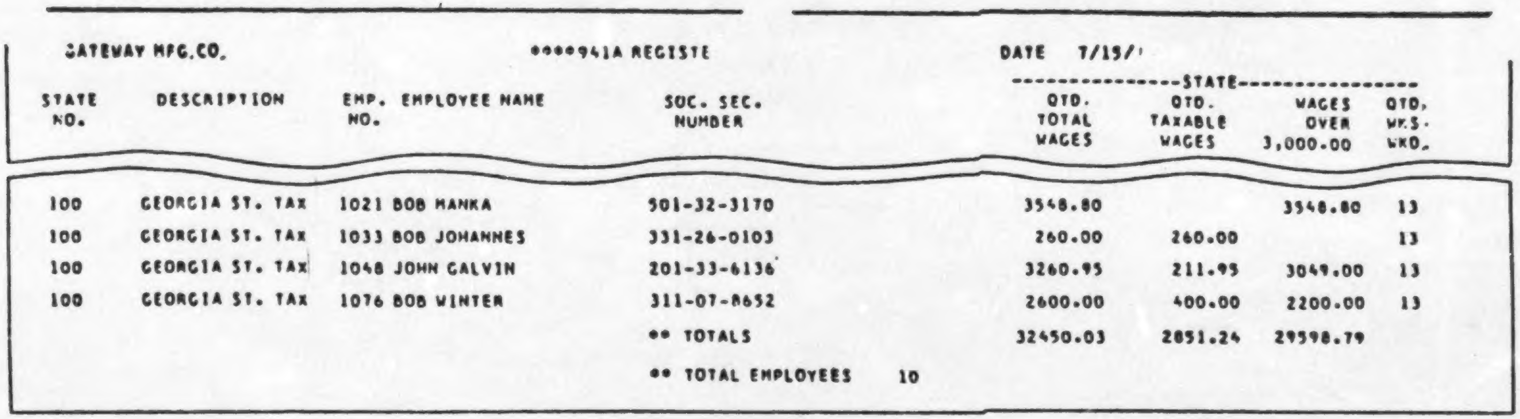
Vacation check

Handles nontaxable pay

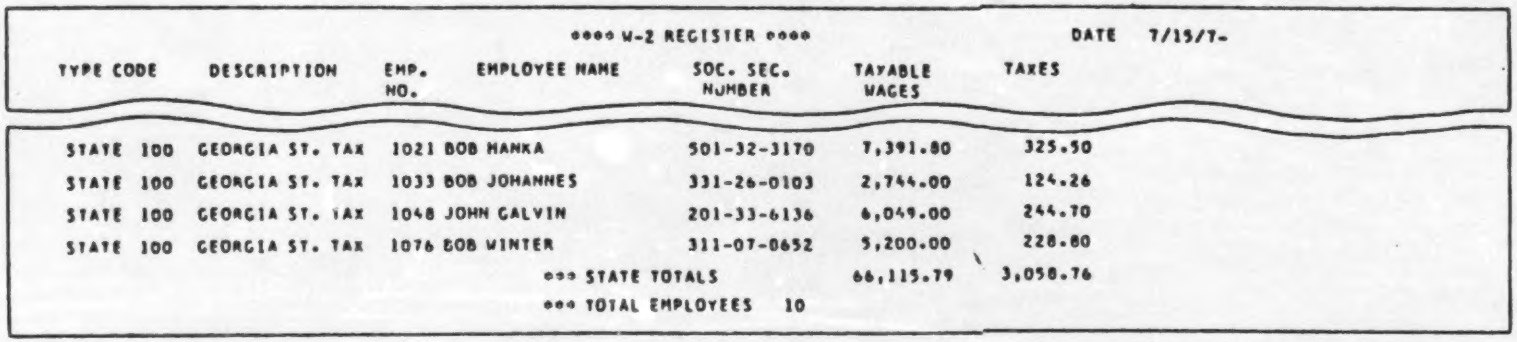
```
GATEWAY MFG.CO.                          ***** PAYROLL REGISTER *****                    DATE  7/15/

EMP   EMPLOYEE NAME      HOURS  REG/OTH  TAXABLE NON-TAX  TOTAL   TAXABLE FEDERAL  FICA     STATE  LOCAL  UNION   MISC     NET     CHE
NO.                              GROSS     ADJ     ADJ    GROSS    GROSS    TAX     TAX      TAX    TAX    DED     DED      PAY      NO

 130 EDWIN DAUM          42.00  251.45                    251.45  251.45   32.70  14.71      6.26          2.50   10.00   185.28    10
 150 JUDY VANDER VEEN    42.00  239.30                    239.30  239.30   42.55  14.00      8.52          2.50  165.00     6.73    10
 270 TOM RYAN            40.00  508.00                    508.00  508.00  155.67  29.72     24.40                         298.21    10
                         40.00  508.00                    508.00  508.00  155.67  22.47     24.40                  22.50   282.96    10
1021 BOB HANKA           42.00  280.65                    280.65  280.65   41.49  16.42      7.82         17.50    5.00   192.42    10
1033 BOB JOHANNES        40.00  100.00    15.00           115.00   15.00            .88                    2.50    5.00   106.62    10
1048 JOHN GALVIN         42.50  255.68                    255.68  255.68   27.73  14.96      5.77          7.50   16.00   183.72    10

* DEPT 22 TOTALS        288.50            15.00         2,158.08          455.81            77.17         32.50          1,255.94
                              2,143.08                          2,058.08          113.16                         223.50

* FINAL CONTROL TOTALS *
---------------- HOURS ----------------      JOB     JOB    --------- ADJUSTMENT -------   ---- DEDUCTION ---- -- UNION ADJUSTMENT ---
  REG.     OTH.    VAC.    SICK    TOTAL    HRS.   GROSS   TAXABLE   NON-TAX     SICK    ONE-TIME  PAY ADV.  TAXABLE     NON TAXABLE

314.50    8.00    40.00   40.00   402.50   242.50 1672.99   15.00      .00        .00     23.00    155.00      .00            .00

------ GROSS ------         SICK      ---------------- TAXES -------------     MISC.       UNION     NET     REVERSE   --- NUMBER --
TOTAL     TAXABLE          GROSS        FIT      FICA     STATE    LOCAL   DEDUCTION     DED TOT   PAY      AMOUNT   EMP,  TRAN.CHE.

2,926.71   2,826.71        100.00     625.67    158.12    95.14      .00     260.72       37.50  1,749.56     .00      9    72     1
```

BEST COPY AVAILABLE

Federal Reports



```
GATEWAY MFG.CO.                 *****941A REGISTE                    DATE   7/15/'
                                                                    ---------------STATE---------------
STATE   DESCRIPTION        EMP.  EMPLOYEE NAME        SOC. SEC.           QTD.       QTD.       WAGES   QTD.
 NO.                        NO.                       NUMBER             TOTAL     TAXABLE      OVER   WKS.
                                                                         WAGES      WAGES    3,000.00  WKD.

 100    GEORGIA ST. TAX   1021 BOB HANKA             501-32-3170        3548.80               3548.80   13
 100    GEORGIA ST. TAX   1033 BOB JOHANNES          331-26-0103         260.00     260.00              13
 100    GEORGIA ST. TAX   1048 JOHN GALVIN           201-33-6136        3260.95     211.95    3049.00   13
 100    GEORGIA ST. TAX   1076 BOB WINTER            311-07-8652        2600.00     400.00    2200.00   13
                                                     ** TOTALS         32450.03    2851.24   29598.79

                                                     ** TOTAL EMPLOYEES    10
```

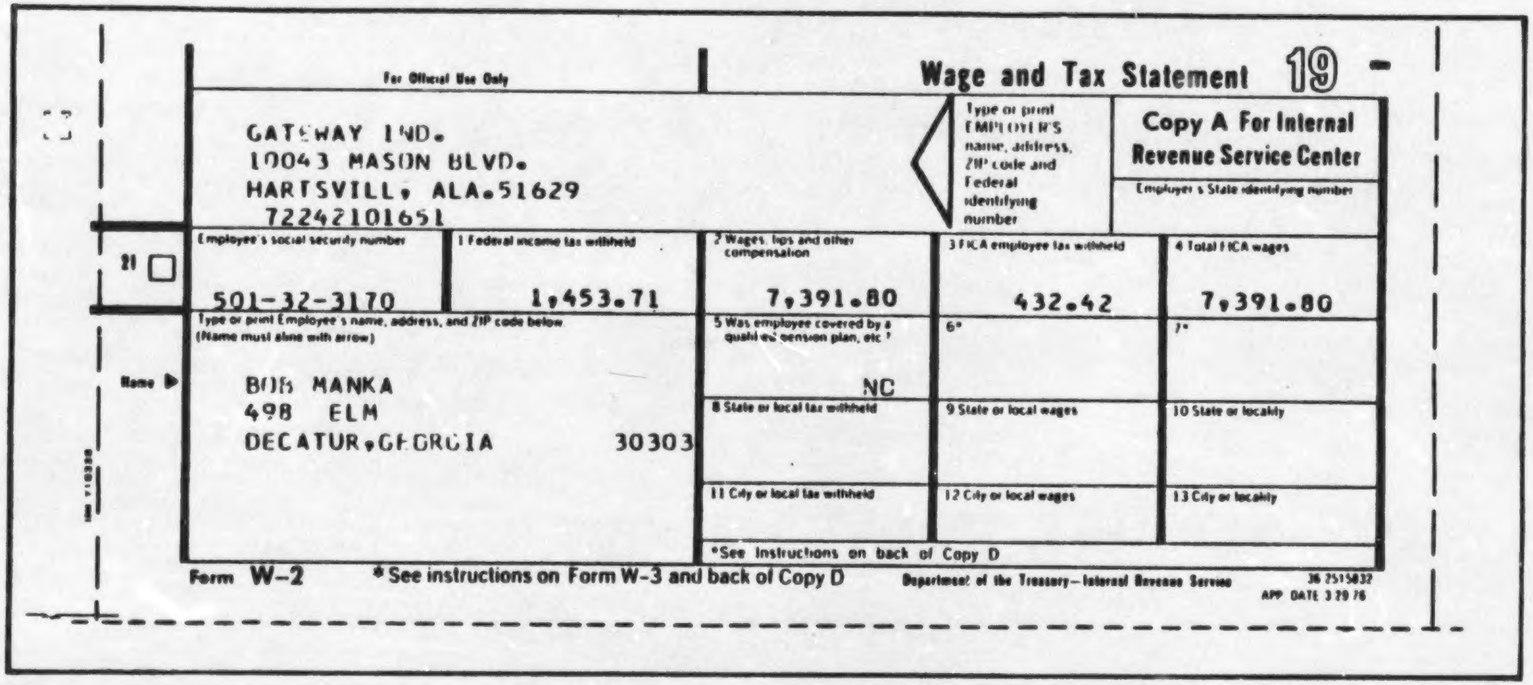


```
                                   **** W-2 REGISTER ****                DATE   7/15/7-
  TYPE CODE     DESCRIPTION       EMP.    EMPLOYEE NAME       SOC. SEC.     TAXABLE      TAXES
                                  NO.                          NUMBER        WAGES

  STATE  100  GEORGIA ST. TAX   1021 BOB HANKA               501-32-3170    7,391.80     325.50
  STATE  100  GEORGIA ST. TAX   1033 BOB JOHANNES            331-26-0103    2,744.00     124.26
  STATE  100  GEORGIA ST. TAX   1048 JOHN GALVIN             201-33-6136    6,049.00     244.70
  STATE  100  GEORGIA ST. TAX   1076 BOB WINTER              311-07-0652    5,200.00     228.80
                                       *** STATE TOTALS                    66,115.79   3,058.76
                                       *** TOTAL EMPLOYEES    10
```

BEST COPY AVAILABLE

HO 14

Employee's Wages and Tax Statement



For Official Use Only

Wage and Tax Statement 19 -

GATEWAY IND.
10043 MASON BLVD.
HARTSVILL, ALA.51629
72242101651

Type or print EMPLOYER'S name, address, ZIP code and Federal identifying number

Copy A For Internal Revenue Service Center

Employer's State identifying number

21 ☐

Employee's social security number 501-32-3170	1 Federal income tax withheld 1,453.71	2 Wages, tips and other compensation 7,391.80	3 FICA employee tax withheld 432.42	4 Total FICA wages 7,391.80

Type or print Employee's name, address, and ZIP code below (Name must align with arrow)

Name ▶
BOB MANKA
498 ELM
DECATUR,GEORGIA 30303

5 Was employee covered by a qualified pension plan, etc.? NC	6*	7*
8 State or local tax withheld	9 State or local wages	10 State or locality
11 City or local tax withheld	12 City or local wages	13 City or locality

*See Instructions on back of Copy D

Form **W-2** * See instructions on Form W-3 and back of Copy D Department of the Treasury—Internal Revenue Service

36 2515032
APP DATE 3 29 76

757

BEST COPY AVAILABLE

Miscellaneous Payroll Reports

```
                                          ****  LABOR DISTR.. TION REGISTER  ****        DATE 7/15/

         WORK     EMP.                  REG.     O/T      OTHER       REG           O/T       OTHER                  G/L     RUN
DEPT.    CTR.     NO.                   HOURS    HOURS    HOURS       GROSS         GROSS     GROSS      ADJUST      NUMBER  CODE
BENCH

22                 130                   8.00                         46.00                                          5090    R
                   150                                                                                   155.00      4210    R
                                         8.00                         44.64                                          4210    R
                   270                  40.00                        508.00                                          4210    R
                                         8.00                        101.60                                          4210    R
                  1021                                                                                    15.00      4210    R
                                         8.00                         52.00                                          4210    R
                  1033                                     40.00                                100.00               4210    R
                                                                                                          15.00      4210    R
                  1048                                                                                     3.00      4210    R
                                                                                                           5.00      4210    R
                                         8.00                         46.64                                          4210    R
      * WORK CENTER TOTAL               80.00              40.00     798.88                     100.00   193.00

      02           130                   4.00                         23.00                                          4210    R
                  1021                   7.25                         47.13                                          4210    R
                                          .75                          4.88                                          4200    S
                  1048                   3.00                         17.49                                          4200    R
      * WORK CENTER TOTAL               15.00                         92.50

                                                  2.00                              17.49
```

• Run re setup

HO 15

BEST COPY AVAILABLE